# Quantum Turing Machines and Quantum Prover-Verifier Interactions

by

Abel Molina Prieto

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science (Quantum Information)

Waterloo, Ontario, Canada, 2020

**Examining Committee Membership**

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

| | |
|---|---|
| External Examiner: | Simon Perdrix<br>Chargé de Recherche,<br>Institut des Sciences de l'Information et de Leurs Interactions,<br>Centre National de la Recherche Scientifique |
| Supervisor: | John Watrous<br>Professor, Cheriton School of Computer Science,<br>University of Waterloo |
| Internal Member: | Richard Cleve<br>Professor and IQC Chair, Cheriton School of Computer Science,<br>University of Waterloo |
| Internal-External Member: | David Gosset<br>Associate Professor, Department of Combinatorics & Optimization,<br>University of Waterloo |
| Other Member(s): | Ashwin Nayak<br>Professor, Department of Combinatorics & Optimization,<br>Cross-Appointed Faculty, Cheriton School of Computer Science,<br>University of Waterloo |

## Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

**Abstract**

We present results on quantum Turing machines and on prover-verifier interactions.

In our work on quantum Turing machines, we continue the line of research opened by Yao (1993), who proved that quantum Turing machines and quantum circuits are polynomially equivalent computational models: $t \geq n$ steps of a quantum Turing machine running on an input of length $n$ can be simulated by a uniformly generated family of quantum circuits with size quadratic in $t$, and a polynomial-time uniformly generated family of quantum circuits can be simulated by a quantum Turing machine running in polynomial time. We then first revisit the simulation of quantum Turing machines with uniformly generated quantum circuits, and present a variation on the simulation method employed by Yao together with an analysis of it. This analysis reveals that the simulation of quantum Turing machines can be performed by quantum circuits having depth linear in $t$, rather than quadratic depth, and can be extended easily to many variants of quantum Turing machines, such as ones having multi-dimensional tapes. Our analysis is based on an extension of a method of Arrighi, Nesme, and Werner (2011) that allows for the localization of causal unitary evolutions, involving abstract lemmas that might be of independent interest.

We also consider the more complex extension of our variant to the circuit simulation of multi-tape quantum Turing machines, where our variant provides a circuit with $O(t^k)$ size and $O(t^{k-1})$ depth for the simulation of $t$ steps of a machine with $k$ tapes. This can be contrasted with the $O(t^k)$ depth corresponding to the generalization of Yao's simulation by Nishimura and Ozawa (2002). Our usage of abstract techniques regarding the localization of causal unitary evolutions allows again for a simplification of the algebraic manipulation aspects of the construction. We also discuss the further extension to the case of oracle quantum Turing machines.

In our work on prover-verifier interactions, we first consider a protocol under the name of perfect/conclusive quantum state exclusion. This means to be able to discard with certainty at least one out of $n$ possible quantum state preparations by performing a measurement of the resulting state. When all the preparations correspond to pure states and there are no more of them than their common dimension, it is an open problem whether POVMs give any additional power for this task with respect to projective measurements. This is the case even for the simple case of three states in three dimensions, which is discussed by Caves, Fuchs and Schack (2002) as unsuccessfully tackled. In our work, we give an analytical proof that in this case POVMs do indeed not give any additional power with respect to projective measurements. We also discuss possible generalizations of our work, including an application of Quadratically Constrained Quadratic Programming that might be of special interest.

We additionally consider the problem of *quantum hedging*, a particular kind of quantum correlation that arises between parallel instances of prover-verifier interactions. M. and Watrous (2012) studied a protocol that exhibited a perfect form of quantum hedging, where the risk for the prover of losing a first game can completely offset the corresponding risk for a second game. We take a step towards a better understanding of this hedging phenomenon by giving a characterization of the prover's optimal behavior for a natural generalization of this protocol. Furthermore, we discuss how the usage of the logarithmic utility principle to analyze prover-verifier interactions could justify further study of quantum hedging.

**Acknowledgements**

The research presented in this thesis was conducted under the supervision and with the collaboration of John Watrous. Thanks to him are due for numerous insightful conversations, without which this research would not have been possible. I thank as well the home members of the committee Richard Cleve, David Gosset and Ashwin Nayak for their time and feedback during my completion of the PhD program. Thanks for his time and willingness to participate in the defense procedure are due as well to the external examiner Simon Perdrix.

This research did also benefit from feedback and conversations with Juani Bermejo Vega, Jonathan Buss, Andrea Coladangelo, Alessandro Cosentino, Ronald de Wolf, Philippe Faist, Alex B. Grilo, Nicolás Guarín-Zapata, Stacey Jeffery, Nathaniel Johnston, Artem Kaznatcheev, George Knee, Robin Kothari, Debbie Leung, Sanketh Menda, Alexandre Nolin, Christopher Perry, Jitendra Prakash, Daniel Puzzuoli, Burak Şahinoğlu, Luke Schaeffer, Jamie Sikora and Jon Tyson, as well as coauthors Srinivasan Arunachalam and Vincent Russo, and anonymous referees.

The work presented here was partially conducted during visits hosted by the Center for Quantum Technologies in Singapore and the Institut de Recherche en Informatique Fondamentale in Paris. Thanks are due to hosts Rahul Jain and Frédéric Magniez, as well as to Ashwin Nayak and the rest of the team handling the regular collaboration between the Institute for Quantum Computing and these host institutions.

This research was funded through the Natural Sciences and Engineering Research Council of Canada, the Mike and Ophelia Lazaridis Graduate Fellowship program, the David R. Cheriton Graduate Scholarship program and the University of Waterloo President's Graduate Scholarship program.

# Table of Contents

**List of Figures** viii

**1 Introduction** 1
- 1.1 Quantum Turing machines . . . 1
- 1.2 Quantum prover-verifier interactions . . . 3
- 1.3 Summary of results . . . 4
  - 1.3.1 Quantum Turing machines . . . 4
  - 1.3.2 Prover-verifier interactions . . . 5
- 1.4 Notation . . . 5

**I Quantum Turing machines** 7

**2 Causality and locality** 8
- 2.1 Setting . . . 9
- 2.2 Results . . . 10

**3 Single-tape quantum Turing machines** 16
- 3.1 Deterministic Turing Machines . . . 16
  - 3.1.1 Definition . . . 17
  - 3.1.2 The classic Boolean circuit simulation of deterministic Turing machines 17
- 3.2 Quantum Turing machines . . . 22

3.2.1 Definition . . . 22
3.2.2 Looped-tape quantum Turing machines . . . 25
3.2.3 Other variants of quantum Turing machines . . . 27
3.3 A variant of the simulation of quantum Turing machines by a quantum circuit 27
3.3.1 Registers in the simulation . . . 28
3.3.2 Operators in the simulation . . . 31
3.3.3 Locality and parallelism . . . 32
3.3.4 Behaviour of the local gate $G$ . . . 35
3.3.5 Recapitulation of the simulation procedure . . . 37
3.3.6 Complexity analysis . . . 37
3.3.7 Differences with Yao's original simulation . . . 41
3.3.8 Sensitivity to model choice . . . 43
3.4 Equivalence between unitarity and isometricity for QTM evolution operators 46
3.4.1 Setting . . . 46
3.4.2 Result and proof . . . 47
3.4.3 Generalizations . . . 48

**4 Multi-tape quantum Turing machines 49**
4.1 Setting . . . 49
4.2 Extension to standard multi-tape quantum Turing machines of our variant for the simulation of quantum Turing machines . . . 51
4.2.1 Setup for extension . . . 51
4.2.2 Obstacle to naive proof for the extension . . . 55
4.2.3 Making the extension work . . . 56
4.2.4 Parallelism and complexity . . . 58
4.3 Oracle quantum Turing machines . . . 62
4.3.1 Definition . . . 62
4.3.2 A first circuit simulation . . . 64

4.3.3 More complex circuit simulations with more standard oracle gate models . . . 67
4.3.4 Other models and further work . . . 76

**II Quantum prover-verifier interactions 78**

**5 Quantum state exclusion 79**
5.1 Setting . . . 79
5.2 Main derivation . . . 82
5.2.1 Restrictions that can be imposed without loss of generality on POVMs that achieve perfect exclusion . . . 82
5.2.2 Verification that any states perfectly excluded by our parametrized optimal POVM satisfy the Caves-Fuchs-Schack inequality . . . 84
5.3 Perspectives for generalization . . . 87
5.3.1 Usage of Quadratically Constrained Quadratic Programs (QCQPs) 87
5.3.2 Direct generalizations of our proof . . . 90
5.3.3 Other considerations . . . 91

**6 Quantum hedging 94**
6.1 Setting . . . 94
6.1.1 Background and motivation . . . 94
6.1.2 Semidefinite programming formulation . . . 98
6.2 Main result . . . 99
6.2.1 Formal statement of main result . . . 99
6.2.2 Proof of Theorem 3 . . . 101
6.3 A motivation from mathematical finance for further study of quantum hedging 110
6.3.1 The logarithmic utility principle . . . 111
6.3.2 Payoff regimes that encourage quantum hedging when two parallel instances of a protocol are conducted in parallel . . . 112

**References 120**

# List of Figures


3.1 The information corresponding to cell $i$ in the Turing machine at any given time step can be written as a function of the information corresponding to the previous time step for cells $\{i-1, i, i+1\}$. We represent with a box $f$ the function that computes the content of $(\mathsf{S}_i, \mathsf{T}_i)$ given the previous contents of $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, and $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$. . . . . 19

3.2 In the simulation of one step in a Turing machine computation, each register pair is updated in parallel. This update is described by the same function $f$ for all register pairs. . . . . 20

3.3 3 steps in the simulation of a classical Turing machine, illustrating the connectivity pattern between the layers in the simulation. . . . . 21

3.4 Each step of a quantum Turing machine computation is simulated by an identical circuit layer, with a general pattern of unitary operations as illustrated here. Observe however that if $N$ is not divisible by 3, one must have one or two additional $G$ operators on separate levels, so that $G$ is applied once to each triple of adjacent register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$. Alternatively, one can increase $N$ to the next multiple of 3 without affecting the performance or correctness of the simulation. . . . . 29

6.1 Illustration of the quantities involved in Theorem 3, for the case where $n = 2$. 100

6.2 Below the curve, we have the region of the $(p, d/x)$ plane where perfect hedging in our 1-out-of-2 scenario is worth it under the logarithmic utility principle. . . . . 114

# Chapter 1

# Introduction

Quantum information processing has received an increasing amount of attention over the last decade, with the development of a large ecosystem of research centers and commercial ventures, and meaningful breakthroughs both regarding theoretical and experimental concerns (see e.g. [57, 70, 96] as recent examples of the former, and [24, 52] as recent examples of the latter).

In the research presented in this thesis, we aim to contribute to progress in quantum information processing through theoretical study. This study centers on two concepts: quantum Turing machines and quantum prover-verifier interactions. We present in Part I the aspects regarding quantum Turing machines, and in Part II the aspects regarding quantum prover-verifier interactions. Part I partially overlaps with the material published in [78], while Part II partially overlaps with the material published in [10] and [76]. An introduction to the results presented in each of the parts follows.

## 1.1 Quantum Turing machines

Turing machines [106] are an elegant abstraction that is key to the fundamentals of computing theory. Therefore, quantum variants of the Turing machine model were sought after and studied in the early days of quantum information processing [35, 22, 23]. This led to a model where the transition function has a similar structure to the transition function for classical Turing machines, but now allows for taking transitions in superposition with each other. However, there are non-trivial constraints on the amplitudes for those superpositions, which enforce that the global evolution of the quantum Turing machine be unitary.

The resulting model proved to be cumbersome to handle mathematically, which led to the adoption of the quantum circuit model as the standard for most theoretical work on quantum information processing. This switch was possible due to seminal research from Yao [119], who established that polynomial time quantum Turing machines are equivalent to polynomial time uniformly generated acyclic quantum circuits. In our work, we study quantum Turing machines with the main focus of further improving our understanding of the relationship between quantum circuits and quantum Turing machines.

We present then in Chapter 3 a new variant of the simulation method employed by Yao, together with an analysis of it. This analysis reveals that the simulation of quantum Turing machines can be performed by quantum circuits having depth linear in $t$ (the number of steps to simulate), rather than quadratic depth, and can be extended to variants of quantum Turing machines, such as ones having multi-dimensional tapes. Another positive attribute of our construction is that it is completely explicit – there is a closed formula that will indicate the output for any input to the gates in our circuit. Yao's original construction only specifies such a closed formula for a subset of possible inputs, and then argues for the ability to solve a linear system in order to obtain valid outputs for the remaining cases. We also fill a gap in the literature in terms of detail, since Yao's paper appeared only as an extended abstract in a conference proceedings, with some details of the proofs left to the reader.

We also introduce in Chapter 3 a model of quantum Turing machines with a finite looped tape, which might be of independent interest for further work on Turing machine computations with a bounded time length. Through the usage of this tool, we present a simpler and more abstract proof of a result from Bernstein and Vazirani [23]. This result establishes that a candidate transition function for a quantum Turing machine will induce a global unitary evolution on the quantum Turing machine if and only if it induces a global isometric evolution, despite the corresponding configuration space being infinite-dimensional.

In Chapter 4, we extend the new variant for the circuit simulation of quantum Turing machines to the case of multi-tape quantum Turing machines with $k$ tapes. If $t$ is the number of steps to be simulated, the resulting circuit has size $O(t^{k+1})$ with depth $O(t^k)$, improving on the $O(t^{k+1})$ depth from a previous construction by Nishimura and Ozawa [81]. We also discuss extensions of this circuit simulation of multi-tape quantum Turing machines to the oracle quantum Turing machine model.

Our analysis in both Chapter 3 and Chapter 4 builds upon the generalization of a result from Arrighi, Nesme and Werner [8] that starts with global causality properties of a unitary evolution operator acting on a composite system, and derives from there non-

trivial locality properties for a family of operators that is related to the unitary evolution we start with. Our generalization, presented in Chapter 2, weakens the conditions needed to derive such localizations, and might potentially be of use in other situations beyond the study of quantum Turing machines.

## 1.2 Quantum prover-verifier interactions

The computational model of prover-verifier interactions considers two agents Alice and Bob who exchange a series of messages, with Alice making a final decision out of a finite set of options.

In Chapter 5, we consider a prover-verifier protocol that goes under the name of quantum *state exclusion*. In this protocol, Alice first gives Bob a quantum system. The state of this system is chosen at random between $n$ options $\{\rho_1, \ldots, \rho_n\}$, with corresponding non-zero probabilities $\{p_1, \ldots, p_n\}$. It is unknown to Bob which of the $\rho_i$ was chosen, but he does know the $\{\rho_1, \ldots, \rho_n\}$ and $\{p_1, \ldots, p_n\}$ values characterizing the corresponding distribution. Bob returns an index between 1 and $n$ to Alice, and Alice determines Bob to have won if he returned an index $j$ such that the state was *not* prepared in the state $\rho_j$. Otherwise, Bob loses. When Bob can win with probability 1, we will say that we have *perfect* state exclusion. This state exclusion protocol is connected to quantum communication complexity [90, 67, 49] and to foundational questions in quantum information [30, 93].

In our work in Chapter 5, we establish that when $n = 3$ and the states $\{\rho_1, \rho_2, \rho_3\}$ are pure states in 3 dimensions, the set of state choices that can be excluded perfectly will not change if Bob's actions are restricted to correspond to a projective measurement. This answers a question left open by Caves, Fuchs and Schack [30]. We have additionally considered different approaches through which one might be able to tackle higher-dimensional questions regarding quantum state exclusion, some of which are known [49] to have meaningful implications in quantum communication complexity. Of special interest are our findings on the usage of Quadratically Constrained Quadratic Programming (QCQP) to model the $n$-dimensional case of quantum state exclusion restricted to projections and pure states. QCQP is a type of mathematical optimization formalism that has seen a large number of applications in recent years, but only limited usage so far within the context of quantum information processing. To our knowledge, this is the first time that the state exclusion of quantum states through projections is expressed through a problem in a standard form of a mathematical optimization framework.

In Chapter 6, we present results related to *quantum hedging*, a phenomena where in order to win at least $k$ out of $n$ parallel instances of a quantum prover-verifier interaction

with a binary win/lose outcome, it is advantageous for Bob to correlate behavior between the instances. This phenomena was discovered during previous Master's work [77], and has been shown to be relevant to the study of quantum coin-flipping protocols [43].

In our work in Chapter 6, we provide tight bounds on the optimal chance of hedging whenever $k = 1$ for a natural family of protocols that generalizes an example previously studied in [77]. This completes our understanding of 1-out-of-$n$ hedging for this family of protocols when put together with results previous to this PhD work. We also provide in this chapter a proof-of-concept analysis of the rationality of engaging in perfect hedging through the lens of the logarithmic utility principle, and discuss how this line of thinking provides a motivation for further analysis of quantum hedging.

## 1.3 Summary of results

For ease of navigation, we present here a summary of the key results in this thesis, together with pointers to the location where they might be found:

### 1.3.1 Quantum Turing machines

- Extension to a wider class of systems of the work from [8] regarding locality properties and causal unitary evolutions. Chapter 2.
- Introduction of a finite looped tape QTM model. Section 3.2.
- Description, proof and analysis of a new variant for the simulation of quantum Turing machines by quantum circuits, including a description of the extension of our simulation variant to multi-dimensional tape settings and other similar variants of quantum Turing machines. Section 3.3.
- A simpler proof of a result from [22] regarding the equivalence of isometricity and unitarity for quantum Turing machine evolutions. Section 3.4.
- A generalization of the variant for the simulation of quantum Turing machines to the multi-tape setting. Section 4.2.
- A discussion of tweaks for the parallelization of previous simulation methods, and the compatibility of our simulation variant with oblivious techniques. Sections 3.3.6 and 4.2.4.

- Further generalizations of this variant to multi-tape settings that involve the usage of oracles. Section 4.3.

### 1.3.2 Prover-verifier interactions

- A proof that POVMs are equivalent to projections for the perfect state exclusion of 3 states in 3 dimensions, solving a problem left open in [30]. Section 5.2.
- A formalization of state exclusion through projections through the QCQP mathematical optimization framework. Section 5.3.1.
- A tight bound on the possibility of achieving quantum hedging for a 2-parameter family of 2-message prover-verifier protocols. Section 6.2.
- A proof-of-concept motivation for further study of quantum hedging through the logarithmic utility principle. Section 6.3.

## 1.4 Notation

We use standard quantum information processing notation, as presented in standard texts [59, 79, 114, 118] on the subject. For convenience, we provide now a brief overview of some of the key elements of our notation.

The main mathematical concept of interest in our studies will be finite-dimensional complex Hilbert spaces, denoted by curly capital letters $\mathcal{H}, \mathcal{K}, \mathcal{X}, \mathcal{Y}$, etc. $\mathrm{L}(\mathcal{H}_1, \mathcal{H}_2)$ refers to the set of all linear maps mapping elements of the space $\mathcal{H}_1$ to elements of the space $\mathcal{H}_2$, while $\mathrm{L}(\mathcal{H})$ corresponds to the set of all linear maps mapping elements from the space $\mathcal{H}$ to elements of that same space.

$\mathrm{Im}(A)$ refers to the image of a map $A \in \mathrm{L}(\mathcal{H})$ (i.e. the set of all its possible outputs), while $\mathrm{Supp}(A)$ refers to its support (i.e. the set of all inputs that are *not* mapped to 0). $\mathrm{U}(\mathcal{H})$, $\mathrm{Proj}(\mathcal{H})$, $\mathrm{Herm}(\mathcal{H})$ and $\mathrm{Pos}(\mathcal{H})$ refer to those subsets of $\mathrm{L}(\mathcal{H})$ corresponding to the maps that are unitary, orthogonal projections, Hermitian, and positive-semidefinite, respectively. For two maps $A, B \in \mathrm{L}(\mathcal{H})$, $[A, B]$ refers to the commutator $AB - BA$.

$\mathcal{H}_1 \otimes \mathcal{H}_2$ is the tensor product of the spaces $\mathcal{H}_1$ and $\mathcal{H}_2$. $\mathcal{H}^{\otimes n}$ is the tensor product of $n$ copies of the space $\mathcal{H}$. For example, $\mathcal{H}^{\otimes 3}$ corresponds to $\mathcal{H} \otimes \mathcal{H} \otimes \mathcal{H}$. Similarly, for $u \in \mathcal{H}$, $u^{\otimes n}$ is the element of $\mathcal{H}^{\otimes n}$ corresponding to the tensor product of $n$ copies of $u$.

Given a space $\mathcal{H}_1 \otimes \mathcal{H}_2$, the partial trace $\mathrm{Tr}_{\mathcal{H}_2}$ maps elements of $\mathrm{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ to elements of $\mathrm{L}(\mathcal{H}_1)$. For $A \in \mathrm{L}(\mathcal{H}_1)$ and $B \in \mathrm{L}(\mathcal{H}_2)$, $\mathrm{Tr}_{\mathcal{H}_2}(A \otimes B) = \mathrm{Tr}(B)A$ , which defines by linearity the action of $\mathrm{Tr}_{\mathcal{H}_2}$ on all members of $\mathrm{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$. $\mathrm{Tr}_{\mathcal{H}_1}$ has a symmetric definition, mapping $\mathrm{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$ to $\mathrm{L}(\mathcal{H}_2)$. For $A \in \mathrm{L}(\mathcal{H}_1 \otimes \mathcal{H}_2)$, $A[\mathcal{H}_1]$ is defined as $\mathrm{Tr}_{\mathcal{H}_2}(A)$.

# Part I

# Quantum Turing machines

# Chapter 2

# Causality and locality

In this chapter, we connect causality and locality properties for linear operators, obtaining results that will be useful for the study of quantum Turing machines in Chapter 3 and Chapter 4.

When researchers express in formal terms the evolution of composite systems, sometimes they employ a top-down view with a global evolution that in some ways respects the local structure. In other occasions, they employ a bottom-up view where local evolutions within the composite system are considered, and together, those local evolutions *define* the global evolution. It is valuable then to establish connections between the objects and results corresponding to the bottom-up and top-down approaches.

Our main theorem here (Theorem 1) goes in that direction, and establishes that if one considers a global unitary evolution with certain causality properties, then some non-trivial locality properties hold for a family of operators derived from the global unitary evolution. This theorem generalizes a result in [8] under assumptions that are a particular case of ours, as will be discussed in more detail later. We will use this theorem in Chapter 3 and Chapter 4 in order to obtain local properties of operators involved in the simulation of quantum Turing machines. These local properties will be derived from a definition of quantum Turing machine that focuses on the global properties of its evolution operator. While it is key to the simulations we consider later, this section does not make usage of the concept of quantum Turing machines, and might also potentially be useful in other contexts.

We present in Section 2.1 the basic setting and definitions needed to present Theorem 1, and then begin Section 2.2 with a statement of the theorem. The remainder of Section 2.2

is devoted to a proof of the theorem, and readers that are not interested in the technical details of this proof may safely skip to the next chapter.

## 2.1 Setting

Our setting will correspond to the split of a composite system into three parts. These three parts will be denoted as registers $\mathsf{X}$, $\mathsf{Y}$, and $\mathsf{Z}$, and they will have state spaces corresponding to the Hilbert spaces $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$, which we take to be finite-dimensional.[1] In this setting, we will say that a unitary operator on the whole system is $\mathsf{Y} \to \mathsf{X}$ causal if the state of register $\mathsf{X}$ after the evolution depends only on the states of registers $\mathsf{X}$ and $\mathsf{Y}$ before the evolution, but not that of register $\mathsf{Z}$. Furthermore, we will allow for the option where such a causality relation holds only relative to a subspace of the complete state space. More formally, we have the following definition:

**Definition 1.** *Let $\mathsf{X}$, $\mathsf{Y}$, and $\mathsf{Z}$ be registers having associated Hilbert spaces $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$, respectively, and let $U \in \mathrm{U}(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z})$ be a unitary operator.*

1. *The operator $U$ is $\mathsf{Y} \to \mathsf{X}$ causal if, for every pair of states $\rho, \sigma \in \mathrm{D}(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z})$ satisfying $\mathrm{Tr}_{\mathcal{Z}}(\rho) = \mathrm{Tr}_{\mathcal{Z}}(\sigma)$, one has*

$$\mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U\rho U^*) = \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U\sigma U^*). \tag{2.1}$$

2. *The operator $U$ is $\mathsf{Y} \to \mathsf{X}$ causal on a subspace $\mathcal{V} \subseteq \mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z}$ if, for every pair of states $\rho, \sigma \in \mathrm{D}(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z})$ satisfying $\mathrm{Im}(\rho) \subseteq \mathcal{V}$, $\mathrm{Im}(\sigma) \subseteq \mathcal{V}$, and $\mathrm{Tr}_{\mathcal{Z}}(\rho) = \mathrm{Tr}_{\mathcal{Z}}(\sigma)$, one has*

$$\mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U\rho U^*) = \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U\sigma U^*). \tag{2.2}$$

In our later application of the concepts here to the study of quantum Turing machines, we will apply this definition repeatedly, with different choices each time about which parts of the system fall under $\mathsf{X}$, $\mathsf{Y}$, and $\mathsf{Z}$.

One might ask why are we using a density operator formalism here when our application domain is that of quantum Turing machines, which are typically described in a framework of pure states and unitary operators. The answer to this question is that while the global state of a composite quantum system may be pure, the corresponding states of local subsystems will generally be mixed.

[1] As will be discussed later, Theorem 1 does still apply if one takes $\mathcal{Z}$ to be an infinite dimensional separable Hilbert space.

## 2.2 Results

In our main result here, we examine the situation when we take an operator $X$ that is local to the space $\mathcal{X}$, and then conjugate it by a unitary $U$ that is $\mathsf{Y} \to \mathsf{X}$ causal on a subspace with a certain tensor product structure. We find that what one obtains through this process is an operator that, as far as the causality subspace is concerned, is local to the spaces $\mathcal{X}$ and $\mathcal{Y}$. Under a couple of extra assumptions, the corresponding local operator will in fact be unitary. The formal statement of our main result here is then as follows:

**Theorem 1.** *Let $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ be complex Euclidean spaces, consider orthogonal sets of nonzero projection operators given by*

$$\{\Delta_1, \ldots, \Delta_n\} \subseteq \operatorname{Proj}(\mathcal{X} \otimes \mathcal{Y}), \qquad \{\Lambda_1, \ldots, \Lambda_n\} \subseteq \operatorname{Proj}(\mathcal{Z}), \tag{2.3}$$

*and let*

$$\Pi = \sum_{k=1}^{n} \Delta_k \otimes \Lambda_k. \tag{2.4}$$

*Let $U \in \mathrm{U}(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z})$ be a unitary operator that is $\mathsf{Y} \to \mathsf{X}$ causal on $\operatorname{Im}(\Pi)$. For every operator $X \in \mathrm{L}(\mathcal{X})$, there exists an operator $W \in \mathrm{L}(\mathcal{X} \otimes \mathcal{Y})$ such that*

$$\Pi U^*(X \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U\Pi = \Pi(W \otimes \mathcal{I}_{\mathcal{Z}})\Pi. \tag{2.5}$$

*If, in addition, $X$ is unitary and $[U^*(X \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U, \Pi] = 0$, then $W$ may also be taken to be unitary.*

Note that given a closed-form description of $\Pi$, $U$ and $X$, we can use Equation (2.5) in order to obtain a closed-form description of $W$ (or more precisely, of $W$'s action on elements of $\operatorname{Im}(\Pi)$). This type of analysis will be performed later in Chapter 3, in the context of our application of Theorem 1 to the simulation of quantum Turing machines.

Arrighi, Nesme and Werner obtain in [8] this result in the particular case where the projection $\Pi$ is equal to $\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$. Note that even in that setting, it is possible to find limits to the localization of non-unitary isometries with a causal structure, as discussed in Section 2 of [19].

As a warmup for the proof of Theorem 1, we will first discuss a proof for this case where $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$.

We begin by using the assumption that $U$ is $\mathsf{Y} \to \mathsf{X}$ causal. In particular, let us consider an arbitrary unitary $Z \in \mathrm{U}(\mathcal{Z})$, as well as an arbitrary pure state $s \in \mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z}$. Then, we have that since

$$\operatorname{Tr}_{\mathcal{Z}}(ss^*) = \operatorname{Tr}_{\mathcal{Z}}\left((\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}} \otimes Z)ss^*(I_{\mathcal{X}\otimes\mathcal{Y}} \otimes Z^*)\right), \tag{2.6}$$

$\mathsf{X} \to \mathsf{Y}$ causality gives us that

$$\mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(Uss^*U^*) = \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}\left(U(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)ss^*(I_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)U^*\right), \tag{2.7}$$

which implies since $X$ is local to $\mathsf{X}$ that

$$\langle X\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}}, Uss^*U^*\rangle = \langle X\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}}, U(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)ss^*(I_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)U^*\rangle\,. \tag{2.8}$$

By rearranging terms, this establishes that

$$\langle U^*(X\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U, ss^*\rangle = \langle U^*(X\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U, (\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)ss^*(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)\rangle\,. \tag{2.9}$$

We now write as

$$A = U^*(X\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U$$

the operator whose locality we want to establish. This allows us to write Equation (2.9) as

$$\langle A, ss^*\rangle = \langle A, (\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)ss^*(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)\rangle\,, \tag{2.10}$$

or what is the same after moving $(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)$ and $(I_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)$ to the left-hand side of the inner product,

$$\langle A - (\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)A(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z), ss^*\rangle = 0. \tag{2.11}$$

Since Equation (2.11) does hold for all choices of pure state $s$, it must follow that

$$A - (\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z^*)A(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z) = 0. \tag{2.12}$$

Through basic algebraic manipulation, this is equivalent to the statement that $A$ commutes with $(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z)$, that is to say,

$$[A, \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}}\otimes Z] = 0. \tag{2.13}$$

Since this holds for any unitary $Z$, and these unitaries span all of $\mathrm{L}(\mathcal{Z})$, it follows from standard linear algebra results that

$$A = W\otimes\mathcal{I}_{\mathcal{Z}}, \tag{2.14}$$

for $W\in\mathrm{L}(\mathcal{X}\otimes\mathcal{Y})$. In particular, these results correspond to a simple case of Tomita's Commutant Theorem, and determine that if an operator $A_1$ commutes with the operator $A_2\otimes\mathcal{I}$ for all choices of $A_2$, it must be of the form $\mathcal{I}\otimes A_3$ for some choice for $A_3$.

It is finally clear that if $A$ is unitary, $W$ must be unitary as well in order for Equation (2.14) to hold.

The main issue when coming up with a proof for the more complex setting in Theorem 1 is that one can at the start only choose $s$ to be an arbitrary pure state in $\mathrm{Im}(\Pi)$, which requires some level of care when trying to formulate the exact next steps (since for example, we do not know that the $(X \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})$ operator will necessarily leave the space $\Pi$ invariant). In our approach to prove Theorem 1 that we will now present, we split a generalization of the reasoning for the $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$ case into 2 lemmas. The first of these lemmas establishes a consequence of our causality assumptions in terms of the properties of the operator $U^*(X \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U$. The second of the lemmas uses these resulting properties in a black-box fashion (i.e. without using the knowledge that we are dealing with the product of 3 operators each of which has their own non-trivial properties) in order to obtain the locality characterization that we seek. After both lemmas are established, Theorem 1 will be proved by chaining the two lemmas.

**Lemma 1.** *Let* $\mathsf{X}$*,* $\mathsf{Y}$*, and* $\mathsf{Z}$ *be registers having associated Hilbert spaces* $\mathcal{X}$*,* $\mathcal{Y}$*, and* $\mathcal{Z}$*, respectively, let* $\mathcal{V} \subseteq \mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}$ *be a subspace, and let* $U \in \mathrm{U}(\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z})$ *be a* $\mathsf{Y} \to \mathsf{X}$ *causal unitary operator on the subspace* $\mathcal{V}$*. For every Hermitian operator* $H \in \mathrm{Herm}(\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z})$ *satisfying* $\mathrm{Im}(H) \subseteq \mathcal{V}$ *and* $\mathrm{Tr}_{\mathcal{Z}}(H) = 0$*, and every operator* $X \in \mathrm{L}(\mathcal{X})$*, one has*

$$\langle H, U^*(X \otimes \mathcal{I}_{\mathcal{Y}} \otimes \mathcal{I}_{\mathcal{Z}})U\rangle = 0. \tag{2.15}$$

*Proof.* Note first that the statement is clear when $H = 0$, and assume from this point on that $H \neq 0$.

Let $\Pi$ be a projector on $\mathcal{V}$. Since $H$ is Hermitian, by the Jordan-Hahn decomposition we can write it as $P - Q$, where $PQ = 0$, $P \geq 0$, $Q \geq 0$. Furthermore, it will be the case that $\mathrm{Im}(P) \subseteq \mathrm{Im}(H)$, and therefore $\Pi P = P$. The same will be the case for $Q$. Note also that $\mathrm{Tr}(H) = \mathrm{Tr}_{\mathcal{X}\otimes\mathcal{Y}}(\mathrm{Tr}_{\mathcal{Z}}(H)) = \mathrm{Tr}_{\mathcal{X}\otimes\mathcal{Y}}(0) = 0$.

Therefore, $\mathrm{Tr}(P) = \mathrm{Tr}(Q)$, and for some real constant $c > 0$ (since $H \neq 0$), we can write $P$ and $Q$ as $c\rho$ and $c\gamma$, respectively, with $\rho$ and $\gamma$ being density matrices on $\mathcal{V}$. Furthermore, note that $\mathrm{Tr}_{\mathcal{Z}}(\rho) - \mathrm{Tr}_{\mathcal{Z}}(\gamma) = \frac{1}{c}\mathrm{Tr}_{\mathcal{Z}}(H) = 0$. Then, from our causality assumptions, it will hold that $\mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U^*\rho U) = \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U^*\gamma U)$.

Since

$$\langle U^*(X \otimes \mathcal{I}_{\mathcal{Y}} \otimes \mathcal{I}_{\mathcal{Z}})U, P\rangle = c\,\langle X, \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U^*\rho U)\rangle \tag{2.16}$$

and

$$\langle U^*(X \otimes \mathcal{I}_{\mathcal{Y}} \otimes \mathcal{I}_{\mathcal{Z}})U, Q\rangle = c\,\langle X, \mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(U^*\gamma U)\rangle\,, \tag{2.17}$$

we have then that $\langle U^*(X \otimes \mathcal{I}_\mathcal{Y} \otimes \mathcal{I}_\mathcal{Z})U, P\rangle = \langle U^*(X \otimes \mathcal{I}_\mathcal{Y} \otimes \mathcal{I}_\mathcal{Z})U, Q\rangle$, and we obtain that

$$\langle U^*(X \otimes \mathcal{I})U, H\rangle = \langle U^*(X \otimes \mathcal{I})U, P - Q\rangle = 0. \tag{2.18}$$

□

**Lemma 2.** *Let $\mathcal{W}$ and $\mathcal{Z}$ be finite-dimensional Hilbert spaces, let $\{\Delta_1, \ldots, \Delta_n\} \subseteq \mathrm{Proj}(\mathcal{W})$ and $\{\Lambda_1, \ldots, \Lambda_n\} \subseteq \mathrm{Proj}(\mathcal{Z})$ be orthogonal sets of nonzero projection operators, and let*

$$\Pi = \sum_{k=1}^{n} \Delta_k \otimes \Lambda_k. \tag{2.19}$$

*For every operator $A \in \mathrm{L}(\mathcal{W} \otimes \mathcal{Z})$ such that $\langle H, A\rangle = 0$ for all Hermitian operators $H$ with $\mathrm{Im}(H) \subseteq \mathrm{Im}(\Pi)$ and $\mathrm{Tr}_\mathcal{Z}(H) = 0$, there exists an operator $W \in \mathrm{L}(\mathcal{W})$ such that*

$$\Pi A \Pi = \Pi(W \otimes \mathcal{I}_\mathcal{Z})\Pi. \tag{2.20}$$

*If, in addition, $A$ is a unitary operator and $[A, \Pi] = 0$ (i.e., $A$ and $\Pi$ commute), then there exists a unitary operator $W$ that satisfies (2.20).*

*Proof.* Let $Z \in \mathrm{U}(\mathcal{Z})$ be an arbitrary unitary operator such that

$$[\mathcal{I}_\mathcal{W} \otimes Z, \Pi] = 0. \tag{2.21}$$

Let then $H$ be an arbitrary Hermitian operator $H \in \mathrm{Herm}(\mathcal{W} \otimes \mathcal{Z})$. It follows from (2.21) that we can write

$$\langle H, \Pi A \Pi - (\mathcal{I}_\mathcal{W} \otimes Z)\Pi A \Pi(\mathcal{I}_\mathcal{W} \otimes Z^*)\rangle = \langle K, A\rangle\,, \tag{2.22}$$

where

$$K = \Pi H \Pi - (\mathcal{I}_\mathcal{W} \otimes Z^*)\Pi H \Pi(\mathcal{I}_\mathcal{W} \otimes Z). \tag{2.23}$$

The operator $K$ is Hermitian and it is such that $\mathrm{Im}(K) \subseteq \mathrm{Im}(\Pi)$ and $\mathrm{Tr}_\mathcal{Z}(K) = 0$. Therefore, by our assumptions in Lemma 2 regarding $A$, it holds that the value of the inner product in (2.22) is equal to 0. Since $H$ is an arbitrary Hermitian operator, it must hold that the operator we are taking its inner product with is equal to zero, which can be written as

$$[\mathcal{I}_{\mathcal{W}} \otimes Z, \Pi A \Pi] = 0. \tag{2.24}$$

Note now that for all unitary operators $Z \in \mathrm{U}(\mathcal{Z})$ with a block structure such that $[Z, \Lambda_k] = 0$ for all values of $k$, the condition in (2.21) holds. Therefore (2.24) holds as well. Following a similar logic as in our earlier transition from (2.13) to (2.14), it follows in turn that we can write

$$\Pi A \Pi = \sum_{k=1}^{n} W_k \otimes \Lambda_k, \tag{2.25}$$

with $W_1, \ldots, W_n$ being linear operators on $\mathcal{W}$ such that $W_k = \Delta_k W_k \Delta_k$ for each $k \in \{1, \ldots, n\}$.

If we then consider the operator

$$W = W_1 + \cdots + W_n + (\mathcal{I}_{\mathcal{W}} - \Delta_1 - \cdots - \Delta_n), \tag{2.26}$$

it follows that

$$\Pi(W \otimes \mathcal{I}_{\mathcal{Z}})\Pi = \sum_{k=1}^{n} \Delta_k W \Delta_k \otimes \Lambda_k = \Pi A \Pi, \tag{2.27}$$

where the second equality follows from (2.25).

When $A$ is unitary and $[A, \Pi] = 0$, $A$ is also unitary when restricted to $\mathrm{Im}(\Pi)$. We have then from (2.27) that each operator $W_k$ must be unitary when restricted to $\mathrm{Im}(\Delta_k)$. The operator $W$ defined in (2.26) will then in turn be unitary, which completes our proof of the claims in Lemma 2. □

Note that even if we did not assume that $\{\Delta_1, \ldots, \Delta_n\}$ is a set of mutually orthogonal operators in (2.19), we would still obtain the decomposition for $\Pi A \Pi$ in (2.25). However, then the operator $W$ defined in (2.26) would not necessarily satisfy the equality in (2.27), since the images of the operators $W_1, \ldots, W_n$ might overlap.

Once we have established Lemma 1 and Lemma 2, it is straightforward to proceed with the proof of Theorem 1:

*Proof of Theorem 1.* Consider the operator $A = U^*(X \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})U$, and the space $\mathcal{W} = \mathcal{X} \otimes \mathcal{Y}$. It follows from Lemma 1 that the assumptions of Lemma 2 are satisfied. The consequences of Lemma 2 do imply then Theorem 1. □

While the consideration of finite-dimensional Hilbert spaces is the main concern of this work, we note that Theorem 1 is true even if one allows the space $\mathcal{Z}$ to be an infinite-dimensional separable complex Hilbert space. This follows from the consideration of extended versions of Lemma 1 and Lemma 2, with the operator $H$ in their statement now constrained to be trace-class (and therefore bounded) in addition to Hermitian. Our proofs go through in these extended settings without any additional issues, as a consequence of basic facts such as those determining that unitaries are bounded, trace-class operators are closed under the product and self-adjoint operations, and the inner product of a trace-class operator and a bounded operator will always be finite. As for the facts that allow the transition from (2.24) to (2.25), they also hold in this case where $\mathcal{Z}$ is an infinite-dimensional separable complex Hilbert space (and in even some more general situations), as discussed for example in Chapter 1, Section 3 of [33] and the notes to Chapter VI of [102].

We will finally note that it is possible to obtain a proof for Theorem 1 more similar to our reasoning in the $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$ case. In particular, the reasoning for the $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$ case can be modified so that it starts with an arbitrary $Z \in \mathrm{U}(\mathcal{Z})$ such that $[\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}} \otimes Z, \Pi] = 0$, together with any pure state $s \in \mathrm{Im}(\Pi)$. Then, each step goes through in the process to derive the relation in (2.11). From there, one obtains that

$$\Pi(A - (\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}} \otimes Z^*)A(\mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}} \otimes Z))\Pi = 0, \tag{2.28}$$

which gives us Equation (2.24) after simple algebraic manipulation.

In light of this remark, it is reasonable to consider the motivations for presenting the proof of Theorem 1 as we did in this chapter. The first reason is that the proof here is the one that we did publish in a peer-review journal publication during the PhD research presented in this thesis, before we had phrased a proof for the $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$ case in the way that is presented here after the statement of Theorem 1. The second reason is that it is of independent conceptual interest to present the approach seen here where we split the proof into two lemmas. For the first of these lemmas it is particularly interesting that its statement cannot be trivially derived from any of the equations in the shorter proof, while for the second lemma it is of interest that it is able to use the consequences from the first lemma in a completely black-box fashion. The third reason is that even if we can reach (2.24) by adapting the reasoning for the $\Pi = \mathcal{I}_{\mathcal{X}\otimes\mathcal{Y}\otimes\mathcal{Z}}$ case, it is convenient later in the context of our proof to be able to make usage of the consequences of Lemma 1. In particular, it is convenient to do so if one performs a step-by-step process of elementary algebraic manipulations in order to reach (2.25) from (2.24).

# Chapter 3

# Single-tape quantum Turing machines

This chapter is organized as follows. In Section 3.1, we discuss introductory concepts related to deterministic Turing machines and their simulation by a circuit. In Section 3.2, we discuss quantum Turing machines and their corresponding state spaces. In Section 3.3, we present and analyze the main result for this chapter. This result is a new variant for the simulation of quantum Turing machines by quantum circuits with some positive properties when compared to previous work in the literature. In Section 3.4, we present a new and computationally simpler proof for a known isometricity vs unitarity equivalence concerning the state space evolutions induced by a QTM transition function candidate.

## 3.1 Deterministic Turing Machines

In addition to providing some intuition about the circuit simulation of Turing machines, our explanation here introduces notation that will be useful in the context of both defining quantum Turing machines and describing a variant for their circuit simulation. We also discuss the roadblocks to a trivial extension to the quantum realm of the classic Boolean circuit simulation of deterministic Turing machines.

### 3.1.1 Definition

We will take deterministic Turing machines to use a state set $Q = \{1, \ldots m\}$ and a tape alphabet $\Gamma = \{0, \ldots, k-1\}$. The machine consists of a two-way infinite tape with its cells indexed by the elements of $\mathbb{Z}$. These cells each contain one of the elements of the tape alphabet $\Gamma$. There is also a tape head that is at each time step above one of the cells in the tape and which reads and writes to that cell in the Turing machine computation. A Turing machine computation begins with the tape head over the tape cell indexed by 0, the input to the computation $x$ written in cells 1, ..., $|x|$, and all other cells set to the 0 symbol, which we will refer to as the *blank* symbol [1]. The initial internal state of the head is state 1. At each step, the logic of the Turing machine evolution is specified by a transition function

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{-1, 1\}. \tag{3.1}$$

If the machine is in the state $p \in Q$ before the transition with the head over a cell indexed by $i$ containing the symbol $a$, and

$$\delta(p, a) = (q, b, D), \tag{3.2}$$

then after a step of the Turing machine evolution the head will be in state $q$ and over the cell indexed by $i + D$, while cell $i$ will have been updated to contain the symbol $b$.

### 3.1.2 The classic Boolean circuit simulation of deterministic Turing machines

We examine here the situation where the first $t$ steps of the computation of a Turing machine are to be simulated. Our goal here then is to describe a circuit that simulates $t$ of the machine's steps and which can be computed efficiently from a description of the transition function $\delta$, following the standard construction described for example in [5].

To begin with, one can observe that during the first $t$ steps of the computation of a Turing machine, the head will be in positions indexed by integers in the set $\{-t, \ldots, t\}$.

[1] In order to simplify our basic mathematical framework, we are using here the symbol 0 to denote blanks. In other contexts that deal with computing specific binary functions in a Turing machine, it is generally easier to denote blanks through the symbol $\square$ or #, and use the symbol 0 to denote the logical binary 0. We will do this ourselves in Section 4.3 when considering oracle quantum Turing machines.

We will further assume for simplicity that the input is of length $n \leq t$, which means that all cells outside of the $\{-t, \ldots, t\}$ index range will be blank during all $t$ steps of the computation.[2] We need then only concern ourselves with the $2t + 1$ cells indexed by $\{-t, \ldots, t\}$.

For each cell in the $\{-t, \ldots, t\}$ index range, one will consider two registers in the circuit simulation of the Turing machine computation. One of the registers stores whether the head is or is not over the cell at the moment, and if so the state of the head. The other register stores the value of the symbol that is currently written in that tape cell. More formally, we have the registers

$$\mathsf{S}_{-t}, \ldots, \mathsf{S}_t \quad \text{and} \quad \mathsf{T}_{-t}, \ldots, \mathsf{T}_t \tag{3.3}$$

where registers $\mathsf{S}_i$ and $\mathsf{T}_i$, corresponding to the cell indexed by position $i$, hold a value from the sets $\{0, \ldots, m\}$ and $\{0, \ldots, k-1\}$, respectively. Register $\mathsf{S}_i$ will contain the value 0 when the head is not in the cell indexed by $i$, and will contain a value from the set $Q = \{1, \ldots m\}$ when the head is in the cell indexed by $i$. Register $\mathsf{T}_i$ will simply contain the symbol stored in the cell indexed by $i$. If one wants the circuit simulating a Turing machine to receive the input $x$ as its only input, one can prepend to the rest of our circuit a pre-processing circuit that initializes the $\mathsf{S}_i$ and $\mathsf{T}_i$ registers to the right value. This is the value corresponding to the input $x$ and the fixed initial position and state for the head.

One then proceeds in the circuit by simulating $t$ steps of the computation of a Turing machine, one step at a time. In the simulation of each individual step of the Turing machine computation, each of the registers in (3.3) will be updated simultaneously. The key fact to consider when examining the structure of such an update is the fact that the Turing machine's head moves one cell at a time, and operates in its reading and writing using only the cell below at the beginning of the step. Therefore, for register $\mathsf{S}_i$, updates are necessary only in case the head was in a position corresponding to registers $\mathsf{S}_{i-1}$, $\mathsf{S}_i$, or $\mathsf{S}_{i+1}$ before the step – otherwise, the head was not in the cell indexed by position $i$ before the transition, and it is not going to be there after the transition either. In case that the head was in a position corresponding to registers $\mathsf{S}_{i-1}$, $\mathsf{S}_i$, or $\mathsf{S}_{i+1}$ before the transition, one can just use the function $\delta$ to see whether the $\mathsf{S}_i$ register will take on a nonzero value after the step, and if so what should be the corresponding state. For example, if before the evolution the register pair $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$ stores $(p, a)$, and $\delta(p, a) = (q, b, -1)$, then after

[2] Observe that if this is not the case, a simulation procedure can simply use the fact that the final value of input cells outside the $\{-t, \ldots, t\}$ index range will be equal to their initial value, and the head will never reach them. The simulation procedure can therefore first record the initial value of those cells in $O(n)$ time and space, and then proceed under our assumption.

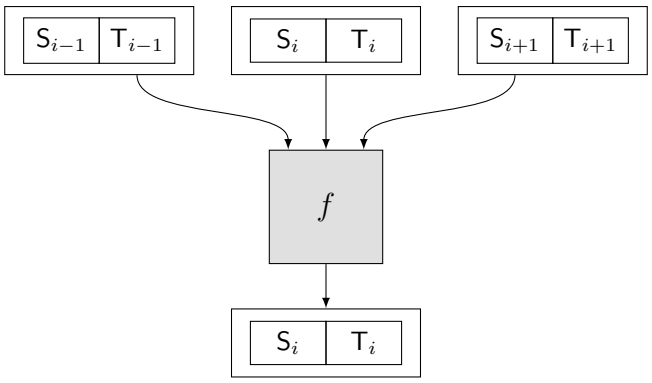


Figure 3.1: The information corresponding to cell $i$ in the Turing machine at any given time step can be written as a function of the information corresponding to the previous time step for cells $\{i-1, i, i+1\}$. We represent with a box $f$ the function that computes the content of $(\mathsf{S}_i, \mathsf{T}_i)$ given the previous contents of $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, and $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$.

the evolution the register $\mathsf{S}_i$ will store the value $q$, representing that the head has moved from position $i+1$ to position $i$ and has state $q$ as its internal state. For $\mathsf{T}_i$, there are only updates to do in case that the head was in position $i$ before the update, and again one can just apply $\delta$ in order compute these updates.

There is then a function $f : \big(\{0,\ldots,m\}\times\{0,\ldots,k-1\}\big)^3 \to \{0,\ldots,m\}\times\{0,\ldots,k-1\}$ that can be computed straightforwardly from $\delta$ and which describes the update rule for the register pair $(\mathsf{S}_i, \mathsf{T}_i)$, given the values before the transition of the three register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, and $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$. The circuit structure of this update is depicted in Figure 3.1. At each time step of the simulation, all $(\mathsf{S}_i, \mathsf{T}_i)$ pairs of registers are updated in parallel, as depicted in Figure 3.2. The one caveat to consider regarding the computation of $f$ is that all inputs $((q_1, a_1), (q_2, a_2), (q_3, a_3))$ to $f$ that arise in the circuit simulation of a Turing machine will be such that at most one out of $q_1$, $q_2$ and $q_3$ will be nonzero, given there is only one head in the machine. On non-valid inputs that represent the existence of more than one head, $f$ can then be taken to have any arbitrary behaviour (similar issues related to the handling of values representing an invalid multi-headed situation will also need to be handled in the quantum case, with less immediate solutions in that case). Note also that at the edges (i.e. when updating the register pairs $(\mathsf{S}_{-t}, \mathsf{T}_{-t})$ and $(\mathsf{S}_t, \mathsf{T}_t)$), the copies of $f$ will be need to be slightly modified so that they hard code the $(0,0)$ values corresponding to the (non-existent) pairs of registers $(\mathsf{S}_{-t-1}, \mathsf{T}_{-t-1})$ and $(\mathsf{S}_{t+1}, \mathsf{T}_{t+1})$.

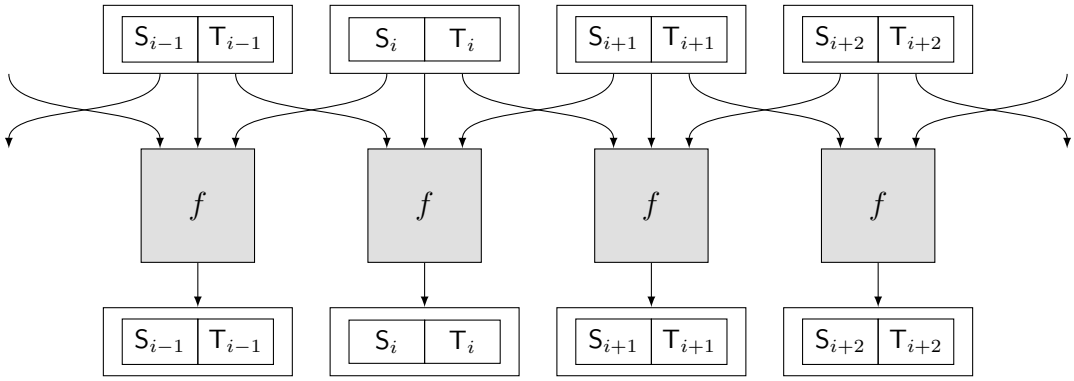


Figure 3.2: In the simulation of one step in a Turing machine computation, each register pair is updated in parallel. This update is described by the same function $f$ for all register pairs.

In order to simulate $t$ steps of a Turing machine computation, $t$ copies of the pattern in Figure 3.2 will be concatenated with each other. This will result in the circuit structure that can be seen in Figure 3.3. Assuming that $|Q|$ and $|\Gamma|$ are constants, $f$ can be represented as a constant-size circuit. This circuit has a regular pattern then such that it can be generated by a deterministic Turing machine with space consumption logarithmic in $t$ (and therefore time consumption polynomial in $t$). In the simulation of a Turing machine computation, this can be preceded by the (also efficiently generated) preprocessing step mentioned earlier, and if one desires, by a postprocessing step that changes the format of the output from that offered by the $(\mathsf{S}_i, \mathsf{T}_i)$ register pairs. Provided that the postprocessing step can be implemented with size $O(t^2)$ and depth $O(t)$ (which is a constraint that allows for a reasonable variety of output formats), the overall circuit will have size $O(t^2)$ and depth $O(t)$.

This construction cannot be easily adapted to the quantum case, where the global evolution of the quantum Turing machine resulting from the local transition function must correspond to a unitary operator. To see why, observe first that there is no exact quantum gate equivalent of the function $f$, since the dimensionality of its input is not equal to the dimensional of its output, and quantum operations in a standard quantum circuit with unitary gates must be reversible (and therefore have equal dimensions of their input and output spaces). One could imagine a solution to this issue that expands the output space for the quantum equivalent of $f$ so that it includes those registers indexed by $i-1$ and $i+1$. However, then we would run into the issue of several potentially non-commuting

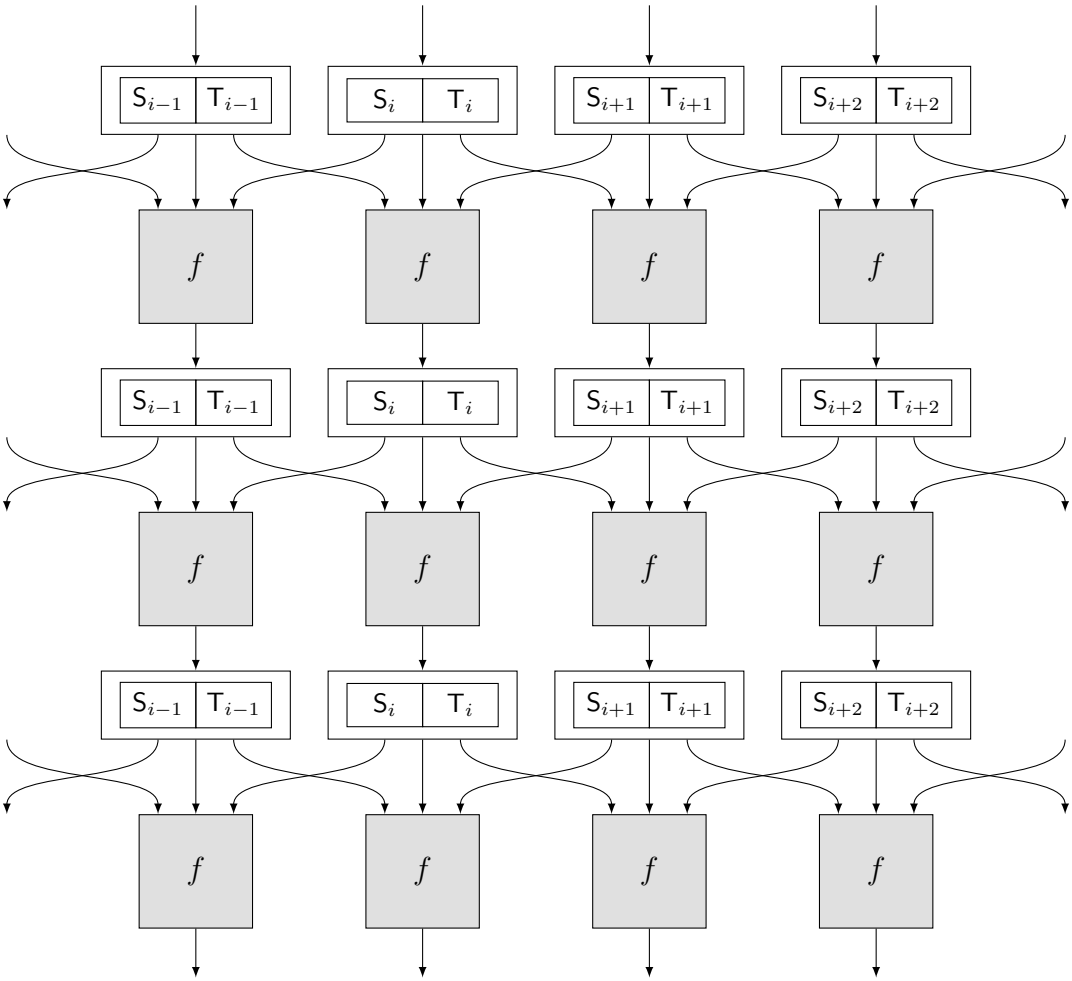


Figure 3.3: 3 steps in the simulation of a classical Turing machine, illustrating the connectivity pattern between the layers in the simulation.

applications of the operator affecting the same register pair. Also, more subtlety is needed than in the classical case when limiting our study to a finite section of the tape, since a global unitary evolution need not be unitary within the context of a truncated tape (consider for example the case of a simple quantum Turing machine that simply ignores the input and always moves its head to the right). It is nontrivial to overcome these obstacles, and it was only with Yao's seminal work [119] that this was achieved. The key tool for this is the expansion of the state space for the $\mathsf{S}_i$ registers, as will be further

described in Section 3.3.1.

## 3.2 Quantum Turing machines

### 3.2.1 Definition

Quantum Turing machines were first formally considered by Deutsch [35], and later defined and examined in further detail by Bernstein and Vazirani [23]. They generalize deterministic Turing machines by allowing for the possibility of making different choices in a transition in superposition with each other, subject to the constraint that the global evolution of the Turing machine be unitary (and therefore reversible).

More formally, we again take quantum Turing machines to use a state set $Q = \{1, \ldots m\}$ and a tape alphabet $\Gamma = \{0, \ldots, k-1\}$. Similarly, the machine consists of a two-way infinite tape with two components. The first component are the cells, indexed by the elements of $\mathbb{Z}$ and each containing at a time one of the elements of the tape alphabet $\Gamma$. The other component is the single tape head. A Turing machine computation begins again with the tape head over the tape cell indexed by 0, the input to the computation $x$ written in cells $1, \ldots, |x|$, and all other cells set to the 0 symbol, which we will refer to as the *blank* symbol. The initial internal state of the head is state 1.

The fact that different choices of transition can occur in superposition with each other is reflected in the fact that the logic at each step of the Turing machine evolution will now be specified by a transition function of the form:

$$\delta : Q \times \Gamma \to \mathbb{C}^{Q \times \Gamma \times \{-1,+1\}}. \tag{3.4}$$

In this function, $\delta(p, a)$ represents the behavior of the tape head when it is in the state $p$ and over a cell containing the symbol $a$. If $\delta(p, a)[(q, b, D)] = \alpha$, we have that in such a situation $\alpha$ is the amplitude for the machine to write the symbol $b$ to the tape, transition into state $q$ and move in direction $D$. However, **not all** transition functions with the signature described in (3.4) will be valid transition functions for a quantum Turing machine. Instead, as will be described in detail later, there will be constraints on $\delta$ that ensure that the global evolution of the quantum Turing machine remains unitary.

Note that in order to define standard quantum computational complexity classes, one must constraint the complex amplitudes in the output of $\delta$ so that they do not for example encode the answer to undecidable problems. A simple way of doing so is to constraint the

amplitudes to the set $\left\{0, \pm 1, \pm i, \pm \frac{1}{\sqrt{2}}, \pm \frac{i}{\sqrt{2}}\right\}$. A more subtle way is to limit the amplitudes to complex numbers that can be efficiently approximated by rational complex numbers. See [2] for further discussion of this matter. In our case, we will in principle describe a construction that given a function $\delta$ builds a circuit for a simulation of the corresponding quantum Turing machine, and then discuss further in Section 3.3.6 how limits to the amplitudes in $\delta$ will be inherited by the amplitudes in the gates that simulate the quantum Turing machine.

To represent the constraint that $\delta$ induces a global unitary evolution for the quantum Turing machine, we must first introduce the concept of *configuration* of a quantum Turing machine. A configuration will be a specification of the position of the head, its internal state, and the contents of the tape. The position of the head will be an integer $i \in \mathbb{Z}$, while its internal state $q$ will be a member of the set $Q$. The contents of the tape will be represented by a function from $\mathbb{Z}$ to $\Gamma$ that maps each cell index to the corresponding alphabet symbol. This function must belong to a subset $\mathcal{F}$ of all possible such functions. In particular, this subset is that of functions from $\mathbb{Z}$ to $\Gamma$ where only a finite number of integers are mapped to symbols other than the blank symbol 0. This constraint is justified by the fact that we only consider Turing machine computations that run for a finite number of steps $t$, and the computations begin with all cells outside the finite input containing the blank symbol.

The constraint that only a finite number of tape cells contain a non-blank symbol means that the set $\mathcal{F}$ is countable, as can be proved through a standard interleaving argument. Therefore, the set $\mathbb{Z} \times Q \times \mathcal{F}$ of all configurations is countable as well. Their complex linear combinations define then a Hilbert space $\mathcal{H}$. This space will be the *configuration space* for the quantum Turing machine. A candidate transition function $\delta$ will induce an operator $U_\delta$ on $\mathcal{H}$ as will be detailed next, and we will require that this operator be unitary in order for $\delta$ to be a valid QTM transition function.

In order to define the action of $U_\delta$ on $\mathcal{H}$, we first introduce for each value of $i \in \mathbb{Z}$ and $a \in \Gamma$ a transformation that maps a symbol assignment $T \in \mathcal{F}$ to $T_{i,a} \in \mathcal{F}$ such that:

$$T_{i,a}(j) = \begin{cases} a & \text{if } j = i \\ T(j) & \text{if } j \neq i. \end{cases} \tag{3.5}$$

This transformation simply changes the assignment in $T$ so that the cell in position $i$ is now assigned the symbol $a$.

Then, the action of $U_\delta$ on basis states of $\mathcal{H}$ is given by

$$U_\delta \left| i, p, T \right\rangle = \sum_{q,a,D} \delta(p, T(i))[(q, a, D)] \ \left| i + D, q, T_{i,a} \right\rangle \tag{3.6}$$

and is expanded by linearity to the rest of $\mathcal{H}$, with the square brackets in (3.6) denoting vector indexing. The condition that $U_\delta$ be unitary can be turned into more explicit constraints on the amplitudes in $\delta$, as described in [23]. The corresponding conditions on the entries of $\delta$ are not directly relevant to our work, but are stated here for the sake of clarity:

1. The set of vectors $\big\{\delta(p, a) \ : \ p \in Q, \ a \in \Gamma\big\}$ is an orthonormal set.
2. For all $(p_0, a_0, b_0)$ and $(p_1, a_1, b_1) \in Q \times \Gamma \times \Gamma$, it holds that

$$\sum_{q \in Q} \delta(p_0, a_0)[q, b_0, +1] \, \overline{\delta(p_1, a_1)[q, b_1, -1]} = 0. \tag{3.7}$$

At a high level, these conditions check that the columns in the infinite matrix corresponding to $U_\delta$ form an orthonormal set. In an infinite-dimensional context, one would generally also need to check that the rows in the matrix form an orthonormal set, but the structure of $U_\delta$ makes that redundant in this case, as proved in [23]. We will also later give an alternative proof of this fact in Section 3.4.

The reason why these conditions are not directly involved with our work is because we do simply assume that the transition function $\delta$ that is provided to our simulation procedure is such that it corresponds to a quantum Turing machine, and that will be enough to derive that the operators in the circuit simulation correspond to unitary gates. If given a function $\delta$ that does *not* correspond to a quantum Turing machine, one follows the procedure we describe, then the circuit that is generated will have non-unitary gates.

The last aspect of the definition of quantum Turing machines that we must consider is the fact that one needs to specify appropriate stopping conditions when discussing Turing machine computations. There are two main approaches to this matter for quantum Turing machines. The first of these approaches, considered in [35], has periodic measurements that determine whether the computation is to be terminated. The second approach, considered in [23], is to have computations that run for a predetermined number of steps (which can be chosen to be a specific function of the size of the input corresponding to the computation). We follow the ideas in this second approach, and assume that the number of steps $t$ to be simulated is part of the input to the algorithm that builds the circuit simulating a Turing machine.

Note that simulation techniques for quantum Turing machines can also be applied to reversible classical Turing machines, since they form a subset of all quantum Turing machines. As for standard classical Turing machines, the work in [17] considers how to simulate $t$ steps of a standard classical Turing machine that use $s$ cells in their space consumption. It finds that for any $c > 0$, this computation can be simulated by a computation on a reversible Turing machine that takes $O(t^{1+c})$ steps and uses $O(s \log t)$ cells in its space consumption, or alternatively, by a computation that takes $O(t)$ steps and uses $O(st^c)$ cells. The work in [64] establishes that it is also possible to use $O(s)$ cells with a number of steps exponential in $s$, while [28] examines tightness concerns regarding the time vs space tradeoff.

Note too that even if uniformly generated quantum circuits are the standard fundamental model for quantum computing, there are also standard aspects of reasoning about quantum algorithms such as loops and conditionals that match quantum Turing machines more closely than they match quantum circuits. There is then work [86, 87, 109, 110] that seeks to move the standard quantum Turing machine model we study here in a direction more amenable to practical work. This often involves the usage of classical control elements, along the lines of measurement-based quantum computing [26].

### 3.2.2 Looped-tape quantum Turing machines

As we saw before, the configuration space $\mathcal{H}$ corresponding to a quantum Turing machine is infinite-dimensional, since the set $\mathcal{F}$ of valid tape contents is infinite. For simplicity, in Section 3.3 we will then work instead with a looped-tape model for quantum Turing machines when discussing their circuit simulation. Note that this not necessary for our results to go through – informally, one could start using an infinite number of wires in the circuit simulation, use causality tools in the corresponding infinite dimensional spaces as we discussed after the proof of Lemma 2, and then finally argue that it is possible to trim all but a finite number of the wires from the circuit. However, we believe that the simplicity of dealing with finite-dimensional spaces all through our discussion of the simulation procedure makes the usage of looped-tape quantum Turing machines worth introducing. This looped-tope model is also new as far as we know, and it seems plausible that its consideration might be of use for future work in quantum information processing.

In a looped-tape quantum Turing machine, we have a circular tape with $N$ cells, indexed by the elements of $\mathbb{Z}/N\mathbb{Z}$. Since the tape is circular, moving to the right from position $N-1$ takes one to position 0, and moving left from position 0 takes one to position $N-1$. This corresponds to the standard addition of $\pm 1$ to the tape index in $\mathbb{Z}/N\mathbb{Z}$. Therefore, the meaning of the transition function remains the same as for a two-way infinite tape

– for $a \in \Gamma$ and $p \in Q$, $\delta(p, a)$ represents again the behavior of the tape head when it is in the state $p$ and over a cell containing the symbol $a$. If $\delta(p, a)[(q, b, D)] = \alpha$ for $(q, b, D) \in Q \times \Gamma \times \{-1, +1\}$, we have that in such a situation the machine will with amplitude equal to $\alpha$ write the symbol $b$ to the tape, transition into state $q$ and move in direction $D$.

The new configuration space $\mathcal{H}_N$ will be spanned then by all the vectors $|q, i, f\rangle$ corresponding to all the possible different choices of $q \in Q, i \in \mathbb{Z}/N\mathbb{Z}, f \in \Gamma^{\mathbb{Z}/N\mathbb{Z}}$. Since $\mathbb{Z}/N\mathbb{Z}$ is a finite group, this is a finite-dimensional complex vector space, as intended. A transition function $\delta$ will induce an operator $U_{\delta,N}$ on $\mathcal{H}_N$ in the same way that it induces an operator $U_\delta$ on $\mathcal{H}$. As for the conditions that $\delta$ must satisfy in order to be a valid transition function (i.e. for $U_{\delta,N}$ to be unitary), it is not hard to prove that for $N \geq 5$, $U_{\delta,N}$ will be unitary if and only if $U_\delta$ is unitary (a short proof of a similar statement is later given in Lemma 4 along the process of further studying the structural properties of quantum Turing machines). The basic idea behind this is that if we consider two standard basis configurations where the head is in different positions, the results of applying $U_{\delta,N}$ to these two configurations when $N \geq 5$ will overlap in at most one possible head position. This is the same as in the infinite-dimensional case, while for $N = 4$ there can be overlap in up to two head positions.

As for the usage of this model in the simulation of quantum Turing machines, observe that if a quantum Turing machine runs for $t$ steps, all tape squares outside the segment of the tape corresponding to indices $\{-t, \ldots, t\}$ will remain with their initial tape content and never contain the tape head. Furthermore, under the assumption that the input to the computation is of length $n \leq t$ (so that the whole input can be read), all tape squares outside this region will in fact remain blank during the computation. Therefore, the computation will be equivalent to a looped-tape QTM computation on a tape of length $N \geq 2t + 1$. We will then simulate this looped-tape computation in the circuits that will be describe in Section 3.3.

Note finally that in order to achieve our goal of working with finite-dimensional spaces, one might naively suggest the alternative approach of truncating the tape to $2t + 1$ cells. This results in a configuration space whose standard basis corresponds to those tuples $(q, i, f) \in Q \times \{-t, \ldots, t\} \times \mathcal{F}$ where $\mathrm{Supp}(f) \subseteq \{-t, \ldots, t\}$. However, this has the issue that this subspace of $\mathcal{H}$ will generally not be invariant under the action of the operator $U_\delta$. The most immediate method to tackle this problem would be to discard those rows and columns from the matrix for $U_\delta$ that correspond to tape heads or non-blank tape symbols outside of the region indexed by $\{-t, \ldots, t\}$. This does however not work either, since the resulting operator might not be unitary.

### 3.2.3 Other variants of quantum Turing machines

Similar to classical Turing machines, one can consider variants of quantum Turing machines, such as quantum Turing machines with multiple tapes, with tapes having a fixed dimension larger than one, with tape heads that have greater freedom in their movements, and so on. For example, Nishimura and Ozawa [82] alter the standard definition of quantum Turing machine so that the head can remain stationary in addition to moving left or right, and provide an equivalent of the condition in Equation (3.7). In this setting, $\delta$ will be a function of the type:

$$\delta : Q \times \Gamma \to \mathbb{C}^{Q \times \Gamma \times \{-1,0,+1\}}. \tag{3.8}$$

Similarly, one can consider a QTM with $k$-dimensional tapes, where $\delta$ will be a function of the type:

$$\delta : Q \times \Gamma \to \mathbb{C}^{Q \times \Gamma \times \{-1,+1\}^k}. \tag{3.9}$$

Our method will very straightforwardly extend to these and similar variants, as will be discussed in Section 3.3.8. Multi-tape quantum Turing machines are another common variant of quantum Turing machines, also considered in [82]. In the context of multi-tape quantum Turing machines, complications prevent a naive extension of our simulation, but these complications can be overcome with a subtler approach, and our simulation consequently extended, as described in Chapter 4.

## 3.3 A variant of the simulation of quantum Turing machines by a quantum circuit

We present here a variant for the simulation of quantum Turing machine circuits. Our purpose is to describe an efficient constructive procedure that given a Turing machine $M$, an input $x$ with length $n$, and a number of steps $t$, generates a circuit that simulates the action of $M$ on $x$ for $t$ steps. The initial setup for the simulation is similar to that from Yao [119], but there are some differences regarding the local operators involved in the simulation, as discussed in Section 3.3.7.

The key property of the simulation is that a step of a quantum Turing machine computation can be simulated through repeated applications of a gate $G$, whose locality is

established using our abstract causality results in Theorem 1. This gate will be defined in such a way that those applications correspond to commuting operators, leading to the parallelization structure that can be seen in Figure 3.4.

As for the positive aspects of the variant when compared to the standard construction from Yao [119], there are three main such aspects. The first of these is that it relies on tools concerning abstract properties of causal unitary evolutions, rather than on the specific algebraic structure of a quantum Turing machine. This makes it very easy to extend to variants of quantum Turing machines, as demonstrated in Section 3.3.4. The second aspect is that it is fully explicit, in that we characterize the amplitudes for the gates in the circuit through specific polynomials in the entries of the transition function $\delta$, as shown in Section 3.3.7. This contrasts with the standard construction, where a constructive but non-explicit algorithmic procedure to find all the amplitudes is specified, with some parts of its proof of correctness left themselves as a sketch in its (proceedings) proof. Last but not least, the circuit we present has depth $O(t)$ and size $O(t^2)$ for the simulation of $t$ steps of a quantum Turing machine. A direct implementation of the standard construction would in contrast have depth $O(t^2)$ and size $O(t^2)$, but it is possible to bring this down to $O(t)$ depth through modifications that are minor relative to the differences with our construction, as further discussed in Section 3.3.6.

### 3.3.1 Registers in the simulation

We will describe our simulation as if we were dealing with simulating a looped-tape quantum Turing machine with length $N = 2t + 1$ and transition function $\delta$, as defined and justified in Section 3.2.2. We assume $t \geq 2$ so that $N \geq 5$ and the global evolution of the looped-tape quantum Turing machine can be taken to be unitary. We will use the symbol $U$ to refer to the configuration space evolution $U_{\delta,N}$ for this looped-tape quantum Turing machine.

In the circuit that we generate, depicted in Figure 3.4 (and whose gates we will describe later in this Section 3.3), we follow the standard technique discussed in Section 3.1.2 and use a pair of registers $\mathsf{S}_i$ and $\mathsf{T}_i$ to represent each position $i \in \{0, \ldots, N-1\}$ in the (looped-tape) quantum Turing machine. As in the work from Yao [119], we consider an expanded state space for the $\mathsf{S}_i$ registers. Such a expanded state space has basis elements corresponding to the set

$$\{-|Q|, -|Q|+1, \ldots, 0, \ldots, |Q|-1, |Q|\}, \tag{3.10}$$

rather than the set $\{0, \ldots, |Q|-1, |Q|\}$. Non-negative values have the same meaning as in the classical case: 0 represents that the head is not in position $i$, while a positive value

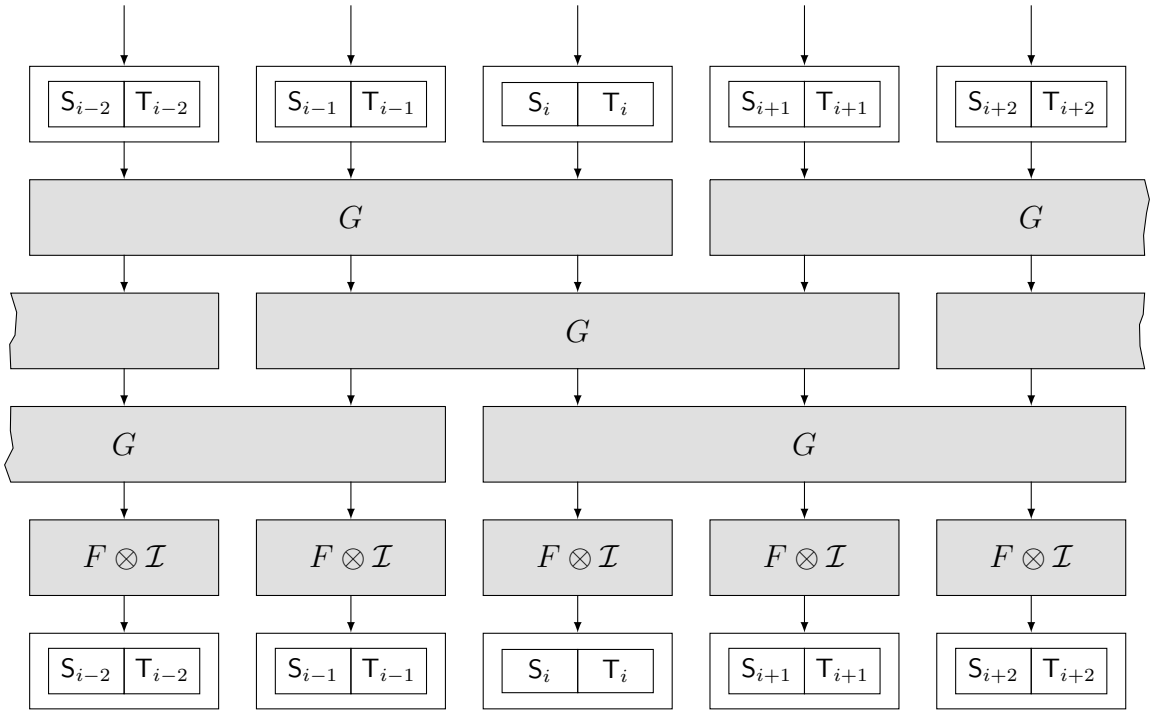


Figure 3.4: Each step of a quantum Turing machine computation is simulated by an identical circuit layer, with a general pattern of unitary operations as illustrated here. Observe however that if $N$ is not divisible by 3, one must have one or two additional $G$ operators on separate levels, so that $G$ is applied once to each triple of adjacent register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$. Alternatively, one can increase $N$ to the next multiple of 3 without affecting the performance or correctness of the simulation.

$s$ represents that the head is in position $i$ and it is in a state $s$. A negative value $-s$ will represent that the head is in position $i$, it is in a state $s$, *and* it has already been moved during this step of the quantum Turing machine simulation and should not be moved again. As for the $\mathsf{T}_i$ registers, the standard basis for its state space is labeled by the elements of the alphabet $\Gamma$.

The state space for the registers $\mathsf{S}_i$ and $\mathsf{T}_i$ will be denoted as $\mathcal{S}_i$ and $\mathcal{T}_i$, respectively. Of course, for different values of $i$ these states spaces will merely be different copies of the same state spaces $\mathcal{S}$ and $\mathcal{T}$. The global state space for all the registers in the circuit will be

$$\mathcal{K}_N = \mathcal{S}_0 \otimes \mathcal{T}_0 \otimes \mathcal{S}_1 \otimes \mathcal{T}_1 \otimes \ldots \otimes \mathcal{S}_{N-1} \otimes \mathcal{T}_{N-1}. \tag{3.11}$$

We now define an isometry $A$ that transforms elements of the state space $\mathcal{H}_N$ for the QTM (see Section 3.2.2 for its definition) into elements of the state space $\mathcal{K}_N$ for the wires in our circuit. This mapping will be essential in Section 3.3.2 in order to algebraically formulate what it means to simulate a step of a QTM with a circuit. We define the mapping based on its action on the standard basis elements of $\mathcal{H}_N$. In particular, if we consider the standard basis element of $\mathcal{H}_N$ associated with the configuration

$$(p, i, T) \in Q \times \mathbb{Z}/N\mathbb{Z} \times \Gamma^{\mathbb{Z}/N\mathbb{Z}}, \tag{3.12}$$

then it will be mapped to the standard basis element of $\mathcal{K}_N$ corresponding to a classical state

$$f(p, i, T) \in (\{0, \ldots, |Q|\} \times \{0, \ldots, |\Gamma| - 1\})^N. \tag{3.13}$$

This classical state $f(p, i, T)$ is determined following the same logic as in the classical simulation described in Section 3.1.2. Therefore, $f(p, i, T)$ will assign to register $\mathsf{T}_j$ a value between 0 and $|\Gamma| - 1$ as determined by $T(j)$. As for the register $\mathsf{S}_j$, it will contain 0 if $j \neq i$, and it will contain the value of $p$ (between 1 and $|Q|$) when $j = i$. No register contains a negative value. Using this definition of $f$, we write $A$ formally as

$$A = \sum_{(p,i,T)} |f(p, i, T)\rangle \langle p, i, T| , \tag{3.14}$$

where the sum iterates over all standard basis elements (i.e. classical configurations) of the configuration space $\mathcal{H}_N$ for the (looped-tape) quantum Turing machine. $AA^*$ will then be a projection into the subspace of the state space $\mathcal{K}_N$ for the registers in the simulation that corresponds to valid quantum Turing machine configurations (i.e. the subspace spanned by the standard basis representations in $\mathcal{K}_N$ of the classical configurations for $\mathcal{H}_N$). In that subspace, there is exactly one $\mathsf{S}_i$ register indicating that the head is there, and the head is not marked as having already moved.

Note finally that one might want to use a different encoding for the input and output configurations in the circuit from that implied by the usage of $\mathsf{S}_0, \ldots, \mathsf{S}_{N-1}$ and $\mathsf{T}_0, \ldots, \mathsf{T}_{N-1}$. If so, such an encoding can be combined without issues with our simulation by appending pre-processing and post-processing circuits, as further discussed in Section 3.3.6.

### 3.3.2 Operators in the simulation

The goal in the circuit simulation is to efficiently implement with quantum gates a unitary evolution that simulates $U$. This can be framed algebraically as implementing a unitary evolution that agrees with $AUA^*$ on $\mathrm{Im}(A)$. This implementation is then concatenated repeatedly to simulate several steps of the quantum Turing machine (i.e. several applications of $U$).

The first of the gates that we will consider corresponds to an operator $F$ acting on $\mathcal{S}_i$ for an arbitrary tape position $i$. This operator performs the mapping

$$F\left|s\right\rangle = \left|-s\right\rangle, \tag{3.15}$$

for $s \in \{-|Q|, -|Q|+1, \ldots, 0, \ldots, |Q|-1, |Q|\}$. This corresponds to a permutation of basis states, and is therefore a valid unitary operator. We will write as $F_i \in \mathrm{L}\left(\mathcal{K}_N\right)$ the operator that acts as $F$ on register $\mathsf{S}_i$ and as the identity on all other registers. Note that $F_0, \ldots, F_{N-1}$ will commute with each other, since they each act in a non-trivial way on a different register. They are also each their own inverse.

In order to define the other gate that we use in the simulation, we must first introduce unitary operators $V$ and $W$ defined as

$$V = AUA^* + (\mathcal{I} - AA^*), \tag{3.16}$$

$$W = (F_0F_1 \ldots F_{N-1})V^*(F_0F_1 \ldots F_{N-1})V. \tag{3.17}$$

$V$ has the purpose of translating the QTM evolution operator $U$ to the state space for the registers in the circuit. $W$ represents an intermediate step in our search for an implementation of this evolution that makes usage of local gates.

It is clear from its definition that $V$ agrees with $AUA^*$ on $\mathrm{Im}(A)$. That is to say, for all values $x \in \mathrm{Im}(A)$, $Vx = AUA^*x$. That means that for all configurations $y \in \mathcal{H}_N$, $V(Ay) = A(Uy)$, and $V$ accurately reflects the result of the QTM evolution.

It also holds that $W$ agrees with $V$ (and therefore with $AUA^*$) on $\mathrm{Im}(A)$. In order to see why, the key insight is that $F_0F_1 \ldots F_{N-1}$ will map elements in $\mathrm{Im}(A)$ to elements in $\mathrm{Im}(A)^\perp$, on which $V^*$ will act as the identity. Therefore, when an element $x \in \mathrm{Im}(A)$ is considered, it holds that

$$Wx = (F_0F_1 \ldots F_{N-1})(F_0F_1 \ldots F_{N-1})Vx = (F_0F_0)\ldots(F_{N-1}F_{N-1})Vx = Vx. \tag{3.18}$$

It follows then that in order to simulate the QTM evolution $U$, it is sufficient to implement a circuit that agrees with $W$ on its action on $\mathrm{Im}(A)$. In order to achieve this, we decompose $W$ into operators suitable to a local implementation. The left-most $F_0F_1 \dots F_{N-1}$ term in $W$ can just be implemented by applying for all tape positions $i$ the gate $F$ to the register $\mathsf{S}_i$. As for the rest of $W$, we rewrite it by adding canceling pairs of $V$ and $V^*$, arriving to the equation

$$V^*(F_0F_1 \dots F_{N-1})V = (V^*F_0V)(V^*F_1V) \dots (V^*F_{N-1}V). \tag{3.19}$$

We then write $V^*F_iV$ as $C_i$. Observe that the operators $C_i$ and $C_j$ will commute with each other for any arbitrary values of $i$ and $j$, as

$$[C_i, C_j] = [V^*F_iV, V^*F_jV] = V^*[F_i, F_j]V = V^*0V = 0. \tag{3.20}$$

$C_i$ and $C_j$ can therefore be implemented in arbitrary order. Next, we will prove in Section 3.3.3 that the operator $C_i$ can be implemented with a local gate $G_i$ that acts on the three register pairs corresponding to positions $i-1$, $i$ and $i+1$ (where the sum of $\pm 1$ is of course performed in $\mathbb{Z}/N\mathbb{Z}$ arithmetic). Since all the $C_i$ operators are the same up to a positional change, they can then be all represented by the same local gate $G$, in a different position each time. This gate $G$ is then the second gate that we will use in the simulation.

Looking back, we can see that the repeated application of the $F$ operator has been introduced in the definition of $W$ in order to later obtain the decomposition in Equation 3.19 and therefore be able to simulate the global evolution $V$ with a composition of local operators. This parallels the application of a similar operator in [8] when looking into the simulation of Quantum Cellular Automata (QCA) by circuits. A similar trick is also used in [42] while discussing the classification of two-dimensional QCA, with its usage being first attributed to unpublished work by Kitaev.

### 3.3.3 Locality and parallelism

Our goal here is to make a successful appeal to Theorem 1 in order to show that for $i \in \{0, \dots, N-1\}$ the operator $C_i$ can be implemented with a local gate $G$ acting on the register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, and $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$.

In our application of Theorem 1, $\mathcal{X}$ corresponds to the state space for the register pair $(\mathsf{S}_i, \mathsf{T}_i)$, while $\mathcal{Y}$ corresponds to the state space for the register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$ and

$(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$, and $\mathcal{Z}$ corresponds to the state space for all other registers. In other words, we have that

$$\mathcal{X} = \mathcal{S}_i \otimes \mathcal{T}_i, \quad \mathcal{Y} = \mathcal{S}_{i-1} \otimes \mathcal{T}_{i-1} \otimes \mathcal{S}_{i+1} \otimes \mathcal{T}_{i+1}, \quad \mathcal{Z} = \bigotimes_{j \notin \{i-1,i,i+1\}} \mathcal{S}_j \otimes \mathcal{T}_j. \tag{3.21}$$

Registers $\mathsf{X}$, $\mathsf{Y}$ and $\mathsf{Z}$ in Theorem 1 are associated then with the registers corresponding to $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$, respectively.

The projector $\Pi$ will project in our application of Theorem 1 into the subspace of $\mathcal{K}_N$ spanned by the basis states where there is exactly one index $i$ such that the register $\mathsf{S}_i$ contains a non-zero value (i.e., the basis states representing only one head, *no matter* whether it has already moved or not). This subspace is left invariant by the application of $V$, and is also left invariant by the application of $F_j$ and $C_j$, for any $j \in \{0, \ldots, N-1\}$. Therefore, states at any point in the execution of our simulating circuit will always belong to that subspace $\mathrm{Im}(\Pi)$. Note that this is not the case if one was to consider the subspace $\mathrm{Im}(A)$. For example, the operator $F_j$ can map some inputs in $\mathrm{Im}(A)$ to outputs in $\mathrm{Im}(A)^{\perp}$, by marking the head as already moved.

As for the decomposition of the projector $\Pi$ with the structure displayed in Equation 2.4 that must exist in order to apply Theorem 1, we write

$$\Pi = \Delta_0 \otimes \Lambda_1 + \Delta_1 \otimes \Lambda_0. \tag{3.22}$$

In this decomposition, for $k \in \{0, 1\}$ we have that $\Delta_k \in \mathrm{Proj}(\mathcal{X} \otimes \mathcal{Y})$ projects into the subspace of $\mathcal{X} \otimes \mathcal{Y}$ representing exactly $k$ heads in registers $\mathsf{S}_{i-1}$, $\mathsf{S}_i$, and $\mathsf{S}_{i+1}$. For $k \in \{0, 1\}$, $\Lambda_k \in \mathrm{Proj}(\mathcal{Z})$ has a similar meaning but now considering all registers $\mathsf{S}_j$ for $j \notin \{i-1,\ i,\ i+1\}$.

The local operator $X \in \mathrm{L}(\mathcal{X})$ in Theorem 1 will correspond to $F$ in our application. As for the unitary operator $U$, it will correspond to $V$ here. We observe that since the head in a quantum Turing machine moves one step at a time, $V$ is $\mathsf{X} \to \mathsf{Y}$ causal relative to $\mathrm{Im}(\Pi)$. That is to say, for $\rho \in \mathrm{D}(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z})$ such that $\Pi\rho\Pi = \rho$ (i.e. such that $\mathrm{Im}(\rho) \subseteq \mathrm{Im}(\Pi)$), it follows from the one-step-at-a-time movement pattern that $\mathrm{Tr}_{\mathcal{Y}\otimes\mathcal{Z}}(V^*\rho V)$ is uniquely determined by $\mathrm{Tr}_{\mathcal{Z}}(\rho)$. Note that this can be false without the assumption that $\Pi\rho\Pi = \rho$. The reason for this is that without assuming that $\mathrm{Im}(\rho) \subseteq \mathrm{Im}(\Pi)$ the behavior of $V$ will depend on the number of registers $\mathsf{S}_j$ with $j \notin \{i-1, i, i+1\}$ that contain a non-zero value, since the action of $V$ depends on whether its input is in $\mathrm{Im}(A)$ or not, by the definition in Equation (3.16).

Finally, $F$ is unitary by definition, and $[V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V, \Pi] = 0$. The commutation relation follows from the fact that $V$, $F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}}$ and $V^*$ are all block diagonal with respect to the blocks defined by $\mathrm{Im}(\Pi)$ and $\mathrm{Im}(\Pi)^\perp$.

We obtain then by the application of Theorem 1 that there is an operator $G \in \mathrm{U}(\mathcal{X}\otimes Y)$ such that

$$\Pi V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V\Pi = \Pi(G \otimes \mathcal{I}_{\mathcal{Z}})\Pi. \tag{3.23}$$

Since $C_i = V^*F_iV = V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V$, this gives us a gate $G$ that can be used to implement the operator $C_i$ as long as inputs in $\Pi$ are concerned, when applied to registers $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1})$, $(\mathsf{S}_i, \mathsf{T}_i)$, $(\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$. As discussed before, only inputs in $\Pi$ will occur during intermediate steps in the application of the operators in $W$, given an input in $\mathrm{Im}(A)$ to $W$. The gate $G$ suffices then for our purpose of simulating $U$ by implementing $W$ (or more exactly, implementing an evolution that agrees with $W$ as far as inputs in $\mathrm{Im}(A) \subseteq \mathrm{Im}(\Pi)$ are concerned). Note that the application of Theorem 1 allow us to find this gate $G$ without directly manipulating any algebraic expressions corresponding the entries of $F_i$ and $V$.

For the overall degree of parallelism for the circuit, we can see in Figure 3.4 that we can completely parallelize the final applications of the $F$ gates, since they all act on different registers. Since the $G$ gates commute with each other and act on three pairs of registers each, we can generally parallelize them into three layers (with up to two additional layers with one operator each in order to deal with boundary cases).

Naively, one might say that for our gates to be geometrically local, we would need to arrange our wires in a circle or cylinder, in order to match the loop structure of the tape. There are however several ways of slightly modifying our construction so that the gates are geometrically local while in a standard linear layout. One way is by flattening such a cylinder, interweaving its two sides after the flattening, and increasing the local neighborhood size from 3 to 5. Alternatively, one can instead use the fact that in a standard quantum Turing machine computation the head begins in the 0 position, and then make this position correspond to cell $\lfloor N/2 \rfloor = t$ in the looped tape. If so, one can add a constant number of wires adjacent to the wires for registers 0 and $M-1$, and keep the circuit as it is, but without implementing the $G$ and $F$ gates corresponding to those new wires. This is valid because the head is never going to reach those positions within $t$ steps, and therefore all the gates in that part of the circuit will just then act trivially in the simulation.

Note that 7-locality for the implementation of $C_i = V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V$ would be considerably easier to establish, relative to the assumption that inputs belong to $\mathrm{Im}(\Pi)$. In brief, the reason for this is that in the product $V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V$, the actions of $V^*$ and $V$

will cancel unless $F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}}$ does not act as the identity. However, since $V$ can only move the head one step at a time, the only way in which this can happen is if the head is in a position corresponding to the range $\{i-1, i, i+1\}$ before the application of $V$. Since $V$ and $V^*$ each move the head one step at a time, this means that the corresponding output of $V^*(F\otimes\mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V$ will have the head in a position corresponding to the range $\{i-3, \ldots, i+3\}$. Furthermore, both $V$ and $V^*$ read/write to only the registers in the simulation within the head's range of movement. Therefore, the product of operators $V^*(F \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V$ will neither modify the registers outside the range $\{i-3, \ldots, i+3\}$, nor use their content in order to determine its action. This 7-locality would not however give us a full conceptual understanding of the operators in our simulation. Furthermore, it would not lead to the explicit expression for the action of $G$ that we present in Section 3.3.4 (it would lead to an algebraically more complex expression), since the computation there implicitly uses the fact that $G$ is known to be 3-local.

### 3.3.4 Behaviour of the local gate $G$

We can easily read off from Equation (3.23) the action of $G$ when it acts on standard basis states. By linearity, this characterizes the action of $G$ on all input states:

1. For input standard basis states of one of the forms

$$|0, a_1\rangle\,|p_2, a_2\rangle\,|0, a_3\rangle\,, \tag{3.24}$$
$$|-p_1, a_1\rangle\,|0, a_2\rangle\,|0, a_3\rangle\,, \tag{3.25}$$
$$|0, a_1\rangle\,|0, a_2\rangle\,|-p_3, a_3\rangle\,, \tag{3.26}$$

   with $p_1, p_2, p_3 \in Q$, $G$ acts as the identity. In the first of these cases, the reason for this is that after $V$ is applied $F$ acts as the identity, and $V^*$ then cancels $V$. In the other two cases, all three of those operators act as the identity.

   We also take $G$ to act as the identity on input standard basis states of the form $|q_1, a_1\rangle\,|q_2, a_2\rangle\,|q_3, a_3\rangle$ where two or more of the values $q_1$, $q_2$, and $q_3$ are nonzero (i.e. standard basis states representing more than one head in the local range that $G$ acts upon and therefore not included in $\mathrm{Im}(\Pi)$). Note that we are free to chose the action of $G$ on these basis states, as long as unitarity is maintained and the space that the states span remains invariant.

2. For input standard basis states of the form $|0, a_1\rangle\,|-p_2, a_2\rangle\,|0, a_3\rangle$, where $p_2 \in Q$, and $a_1, a_2, a_3 \in \Gamma$, $G$ performs the following transformation:

$$\begin{aligned} G : |0, a_1\rangle\,|-p_2, a_2\rangle\,|0, a_3\rangle \mapsto \\ \sum_{\substack{p_1 \in Q \\ b_1 \in \Gamma}} \overline{\delta(p_1, b_1)[p_2, a_1, +1]}\,|p_1, b_1\rangle\,|0, a_2\rangle\,|0, a_3\rangle \\ + \sum_{\substack{p_3 \in Q \\ b_3 \in \Gamma}} \overline{\delta(p_3, b_3)[p_2, a_3, -1]}\,|0, a_1\rangle\,|0, a_2\rangle\,|p_3, b_3\rangle\,. \end{aligned} \tag{3.27}$$

3. For input standard basis states of the form $|p_1, a_1\rangle\,|0, a_2\rangle\,|0, a_3\rangle$ or $|0, a_1\rangle\,|0, a_2\rangle\,|p_3, a_3\rangle$, where $p_1, p_3 \in Q$ and $a_1, a_2, a_3 \in \Gamma$, $G$ performs the following transformation:

$$\begin{aligned} G : |p_1, a_1\rangle\,|0, a_2\rangle\,|0, a_3\rangle \mapsto \\ \sum_{\substack{q_2 \in Q \\ b_1 \in \Gamma}} \delta(p_1, a_1)[q_2, b_1, +1]\,|0, b_1\rangle\,|-q_2, a_2\rangle\,|0, a_3\rangle \\ + \sum_{\substack{q_1 \in Q,\, r_0 \in Q \\ b_1 \in \Gamma,\, c_1 \in \Gamma}} \delta(p_1, a_1)[r_0, c_1, -1]\overline{\delta(q_1, b_1)[r_0, c_1, -1]}\,|q_1, b_1\rangle\,|0, a_2\rangle\,|0, a_3\rangle\,. \end{aligned} \tag{3.28}$$

and

$$\begin{aligned} G : |0, a_1\rangle\,|0, a_2\rangle\,|p_3, a_3\rangle \mapsto \\ \sum_{\substack{q_2 \in Q \\ b_3 \in \Gamma}} \delta(p_3, a_3)[q_2, b_3, -1]\,|0, a_1\rangle\,|-q_2, a_2\rangle\,|0, b_3\rangle \\ + \sum_{\substack{q_3 \in Q,\, r_4 \in Q \\ b_3 \in \Gamma,\, c_3 \in \Gamma}} \delta(p_3, a_3)[r_4, c_3, +1]\overline{\delta(q_3, b_3)[r_4, c_3, +1]}\,|0, a_1\rangle\,|0, a_2\rangle\,|q_3, b_3\rangle\,. \end{aligned} \tag{3.29}$$

We can observe that the amplitudes in $G$ have a fairly straightforward dependencies on the amplitudes for $\delta$ that just involves taking products, sums, and complex conjugations.

Note that in the third case, one might expect from the expression in the left-hand side of (3.23) that there would be terms where the head ends up outside the considered range of registers. For example, in (3.28) we might expect to have a term where the head first goes left with the application of $V$ and then goes left again with the application

of $V^*$. However, we know from our application of Theorem 1 and the corresponding structure of the right-hand side of (3.23) that such terms cannot exist, which means that the corresponding coefficients that would be derived from $\delta$ must be zero for any choice of quantum Turing machine. The correctness of our construction gives then a method to arrive to this knowledge without explicitly using the properties of the entries in $\delta$ determined in [23] and presented in Section 3.2.1. Note that one could also aim to use those properties in order to further simplify the expressions in (3.28) and (3.29), which we have not done in light of our purpose of illustrating how the amplitudes in $G$ can be read off in a simple manner from Equation (3.23).

### 3.3.5 Recapitulation of the simulation procedure

In short, the procedure that we consider in order to simulate with a circuit a quantum Turing machine $M$ for $t$ steps, given a string $x$ of length $n \leq t$, is the following:

1. We simulate an equivalent computation on a looped-tape quantum Turing machine of length $N = 2t+1$. In order to represent the cells in this tape, we use in our circuit $N = 2t + 1$ register tuples $(\mathsf{S}_0, \mathsf{T}_0), \ldots, (\mathsf{S}_{N-1}, \mathsf{T}_{N-1})$.

2. Let $G$ be the gate defined by the entries in $\delta$ as specified in Section 3.3.4. The simulating circuit applies the gate $G$ to each set of three adjacent register pairs $(\mathsf{S}_{i-1}, \mathsf{T}_{i-1}), (\mathsf{S}_i, \mathsf{T}_i), (\mathsf{S}_{i+1}, \mathsf{T}_{i+1})$, where adjacency is determined in $\mathbb{Z}/N\mathbb{Z}$ arithmetic. These applications commute with each other and can be parallelized when they act on non-overlapping sets of registers, as illustrated in Figure 3.4. Then, a copy of the $F$ gate is applied to each of the registers $\mathsf{S}_0, \ldots, \mathsf{S}_{N-1}$. These applications of the $G$ gates and $F$ gates are repeated $t$ times.

3. The final configuration of the machine $M$ after $t$ steps will correspond now to the state of the registers $(\mathsf{S}_0, \mathsf{T}_0), \ldots, (\mathsf{S}_{N-1}, \mathsf{T}_{N-1})$. If desired, an additional circuit can be appended in order to transform the encoding of the output configuration (same applies to prepending a circuit for translating the encoding of the initial configuration).

### 3.3.6 Complexity analysis

The applications of the gate $G$ for one step of the simulation can be parallelized to a small constant number of layers (3 to 5, see caption of Figure 3.4). This follows from

the facts that they all commute with each other and act each on a set of 3 register pairs. Together with the one layer of applications of the gate $F$, this gives us $O(1)$ circuit depth and $O(N) = O(t)$ circuit size for simulating one step of a QTM. Concatenating $t$ times we obtain a circuit with $O(t)$ depth and $O(t^2)$ size.

Regarding the generation of the circuit, we take it that the state set and alphabet for the quantum Turing machine $M$ are of constant size (if they are not, the circuit generation will still run in polynomial time as a function of their size). We also assume that the entries of $\delta$ are provided and of constant size. If one wishes to instead encode the amplitudes for $\delta$ in a non-explicit way, then any negative properties of the computation process for these amplitudes would be inherited by the procedure here that builds the simulating circuit. Related to this concern, note that in order to consider a set of quantum Turing machines that covers the complexity class **BQP**, it is enough to consider those with rational amplitudes, as earlier discussed in Section 3.2.1 and proved in [2]. Note also that if the amplitudes in $\delta$ are restricted to be in a finite set, the amplitudes in $G$ will also similarly be restricted to a finite set. In particular, the structure of $G$ examined in Section 3.3.4 determines that it suffices to consider the finite set obtained by starting with the amplitudes in $\delta$, adding to the set their conjugates, then adding all pairwise products, taking the closure under addition restricted to membership within the complex unit disk, and finally adding the numbers 0 and 1 if needed.

The regularity of our circuit guarantees the uniformity of our construction. More precisely, for any quantum Turing machine $M$ to simulate, there is a deterministic Turing machine with a logarithmic amount of writable space (and a separate output tape) such that on input $1^n01^t$, produces a description of a quantum circuit that simulates $M$ on inputs of length $n$ for $t$ steps. This corresponds to the fact that due to the regularity of the circuit pattern, the bottleneck for space consumption in the circuit generation is simply to store counters about where in the process are we along the vertical and horizontal axis. Note that one could instead consider the specification of $M$ as a constant-sized input to the deterministic Turing machine that builds the circuit, given our assumptions in the previous paragraph.

Retaining the asymptotic performance of the simulation adds constraints to the space of possible alternative encodings that can be used through preprocessing and postprocessing operations. However, these constraints are fairly minor and still allow for a wide variety of encodings. In particular, the complexity of the construction will not change as long as we have (uniform) preprocessing and postprocessing circuits with size $O(t^2)$ and depth $O(t)$, which allows for a wide variety of procedures.

In particular, it is possible to accommodate without changing the simulation the (ar-

guably) most obvious alternative output encoding idea. In this alternative, we express the output through classical configurations for a (looped-tape) quantum Turing machine. These classical members of $\mathcal{H}_N$ can be represented through a function $f$ as a sequence of $N+2$ integers corresponding to $q \in Q$, $i \in \mathbb{Z}/N\mathbb{Z}$, and $T(j) \in \Gamma \in$ for $j \in \mathbb{Z}/N\mathbb{Z}$. We will refer as $\mathcal{G}$ to the state space spanned by the linear combinations of these tuples. It is possible to express the output of our circuit in this format with a circuit of $O(t \log t)$ size and $O(\log t)$ depth. To see why, one can first consider a classical circuit with the desired performance, and then convert this to a reversible circuit using standard techniques (see e.g. [16, 105]).

Such a classical circuit can be obtained by recursively combining the information in the $\mathsf{S}_i$ registers (note that for the content of the tape in the $\mathsf{T}_i$ registers, the encoding does not change). This recursion technique results in a circuit with depth $O(\log N) = O(\log t)$ and size $O(t \log t)$. After making usage of standard reversible techniques, this turns into a reversible circuit with the same complexity that for each standard basis configuration $h \in \mathrm{Im}(A)$, maps $|h\rangle |y\rangle$ to $|h\rangle |y \oplus f(x)\rangle$, where $f$ is the encoding transformation specified in the previous paragraph, $y$ is an arbitrary standard basis member of $\mathcal{G}$, and the $\oplus$ operation is executed bitwise on each of the $N+2$ integers in the tuple. This circuit might also make usage of a $O(t \log t)$ number of auxiliary inputs set to a fixed $|0\rangle$ state and returned to this state after the execution of the circuit. Quickly summarizing those standard techniques, this reversible circuit can be obtained by first expressing each classical gate through reversible (e.g. CNOT or Toffoli) gates, then using CNOT gates to write the answer into the output bits (this is the part that takes $y$ to $y \oplus f(x)$), and finally concatenating in reverse order the reversible gates we used to compute the answer, which sets all other registers back to their initial state.

One might object to the fact that the gate $G$ in our circuit depends on the structure of $\delta$, instead of belonging to a standard gate set. If one is instead interested in a simulation with overall error $\varepsilon$ using a fixed universal set of gates, then each copy of $G$ must be implemented with accuracy on the order of $O(\varepsilon/t^2)$. Implementing $G$ with this accuracy is possible with circuits of size polylogarithmic in $t$ and $1/\varepsilon$, as established (constructively and efficiently) by the Solovay–Kitaev theorem [34, 60]. That will increase the size and depth bounds for our circuits by this polylogarithmic factor. Alternatively, if we assume that arbitrary single-qubit gates can be implemented, then one can build the gate $G$ exactly out of a constant number of CNOT and single-qubit gates (the single-qubit gates will depend on the entries in $\delta$, as does $G$). This follows from the work in [13]. Regarding the implementation of single-qubit gates themselves, it is of interest to consider the main result in [61], which shows how to optimally use Clifford and $T$ gates in order to implement those single-qubit unitaries that have all of their entries in the ring $\mathbb{Z}[\frac{1}{\sqrt{2}}, i]$.

It is also natural to consider possible tradeoffs between circuit size and circuit depth, following the existence of such improvements in the classical context. More in detail, one can classically improve on the circuit corresponding to Figure 3.3, and obtain a circuit size of $O(t \log t)$ [98, 91]. Doing so is non-trivial, and involves a reduction from Turing machines to *oblivious* Turing machines. These are a subcategory of Turing machines with the additional property that the position of the tape head at time $t$ follows a fixed pattern as a function of $t$ that is independent of the input to the machine, given a fixed input length. In order to simulate such a Turing machine by a uniformly generated circuit, one can start with the circuit pattern in Figure 3.3, and then trim all but a constant number of applications of $f$ per time step of the simulation. In particular, if at time $t$ the head is known to be in position $p_t$, one can only keep the three instances of $f$ that write to registers indexed by positions $p_t - 1$, $p_t$ and $p_t + 1$. It is known that for any Turing machine $M$ we can efficiently construct a 2-tape oblivious Turing machine that simulates $t$ steps of $M$ while taking $O(t \log t)$ steps itself. While the construction in Figure 3.3 does not trivially extend to the multi-tape case, it does so in the trimmed case where we assume we know the position of the heads at each time. By simulating in this way the oblivious 2-tape machine with a circuit, one can then obtain uniformly generated circuits that simulate $t$ steps of $M$ with size $O(t \log t)$ and depth $O(t \log t)$ (observe that the bound on the depth increases from the original $O(t)$-depth construction, while the size bound decreases). It is an open problem [1] whether this circuit size can be improved upon.

In the quantum case, Carpentieri has presented [29] a construction for simulating a standard quantum Turing machine with an oblivious 4-tape quantum Turing machine (i.e. a quantum Turing machine where after any step, the machine is in a superposition of classical configurations for which the positions of the 4 heads are all equal to a given predetermined position). Based upon the correctness of the construction in [29], one can then obtain a circuit for the simulation of quantum Turing machines by a quantum circuit with size and depth $O(t \log t)$. This follows from considering that in the multi-tape version of our Turing machine simulation variant (to be presented in Chapter 4), all but a constant number of the applications of $G$ can be removed if the machine to simulate is oblivious and has a constant number of tapes. The reasoning for this follows a similar logic as in the classical case together with the observation that the applications of $G$ commute with each other, and is presented in more detail in Section 4.2.4. Note that the oblivious quantum Turing machine in [29] allows for stationary tape movements, but this minor change does not prevent the application of our simulation technique (see Section 3.3.8 for further discussion of how the steps in our construction and proof go through with a possibly stationary tape head or other similar modifications).

One can also use the techniques in [112] that convert between quantum Turing machines

and a subclass of one-dimensional quantum cellular automata in order to make the position of the tape-heads deterministic, but this would not help us improve on the $O(t^2)$ circuit size. In particular, we can first simulate $t$ steps of a QTM with $O(t)$ steps of a QCA, and then simulate those steps with $O(t^2)$ steps of a QTM such that the head position is deterministic. However, the extra number of steps cancels then the advantage from knowing that the head position is deterministic. These techniques could also be used in an alternative method for obtaining a circuit that simulates $t$ steps of a quantum Turing machine while having $O(t^2)$ size $O(t)$ depth. This would be by first simulating $t$ steps of a QTM with $O(t)$ steps of a QCA, and then composing that with the method in [8] for the simulation of quantum cellular automata.

One final observation regarding the circuit size of our simulation variant is that if one compares with the quantum cellular automata simulation in [8], there is a reduction of the multiplicative constant associated with certain space costs. This corresponds to the replacement in the cellular automata case of the state space $\Gamma$ associated with each cell by the state space $\Gamma^2$ in the corresponding circuit simulation, which is not necessary here (we do however need to explicitly store an element of $\{-|Q|, \ldots, 0, \ldots, |Q|\}$ for each cell). As discussed in Section 4.2.2 of [71], such a saving in half of space consumption is also present in the work published in [7] that looks into the simulation of reversible causal graph evolutions that generalize cellular automata.

### 3.3.7 Differences with Yao's original simulation

There is a gate in Yao's construction that plays a role similar to $G$ here, and it is then of interest to compare $G$ with that gate. The most relevant observation from such a comparison is that in Yao's definition, unlike our case, there is a non-trivial transition out of the situation where the head is on the middle cell out of the 3, and it has not been moved yet, as follows:

$$\begin{aligned} G : |0, a_1\rangle |p_2, a_2\rangle |0, a_3\rangle \mapsto \\ \sum_{\substack{q_2 \in Q \\ b_2 \in \Gamma}} \delta(p_2, a_2)[q_2, b_2, -1] \, |-q_2, a_1\rangle |0, b_2\rangle |0, a_3\rangle \\ + \sum_{\substack{q_2 \in Q \\ b_2 \in \Gamma}} \delta(p_2, a_2)[q_2, b_2, +1] \, |0, a_1\rangle |0, b_2\rangle |-q_2, a_3\rangle \, . \end{aligned} \tag{3.30}$$

(More precisely, Yao does this for QTMs that allow for stationary tape heads, and we present here the simplified form when such transitions are not allowed).

The rest of the action of the gate in Yao's construction (about which we will say more later in this section) is defined first so that there is a certain subspace where the gate acts as the identity, and then an existence proof is provided for the possibility to complete the rest of entries for the corresponding unitary through solving a linear system.

Yao's construction does however involve a different circuit pattern to that in Figure 3.3. In Yao's construction, the corresponding gate $G$ is also being applied to each tuple of three adjacent register pairs, but these operators are being applied in a cascading pattern, rather than in parallel, which does naively leads to a $O(t^2)$ depth with a $O(t^2)$ circuit size in order to simulate $t$ steps of a quantum Turing machine. It is however possible to perform only minor modifications to this construction to achieve parallelism similar to the $O(t)$ depth for the variant we present. One approach to doing this is to parallelize the cascades corresponding to each of the steps in the simulation, similar to pipelining in computer architecture. A second approach is to alter the definition of the subspace where the equivalent of the gate $G$ acts as the identity in order to make this definition symmetrical with respect to the leftmost and rightmost register tuples in the neighbourhood that the gate acts upon, which leads to commutation properties similar to the ones that allow parallelism in our variant.

We now discuss at a high level the reasons that guide the structure of Yao's construction. This discussion will be helpful in Chapter 4 when we consider how does our analysis of parallelism modifications to Yao's construction extend to the multi-tape generalization of the construction.

The main guiding force behind the structure of the definition is that we want the output states obtained in Equation (3.30) not to be perturbed by other copies of $G$ applied later in the simulation. However, we cannot say that $G$ will act as the identity on all input states that represent a head that has already been moved. In particular, $G$ cannot act as the identity on the outputs in (3.30) that are of the form

$$\begin{aligned} &\sum_{\substack{q_2 \in Q \\ b_2 \in \Gamma}} \delta(p_2, a_2)[q_2, b_2, -1] \left|-q_2, a_1\right\rangle \left|0, b_2\right\rangle \left|0, a_3\right\rangle \\ + &\sum_{\substack{q_2 \in Q \\ b_2 \in \Gamma}} \delta(p_2, a_2)[q_2, b_2, +1] \left|0, a_1\right\rangle \left|0, b_2\right\rangle \left|-q_2, a_3\right\rangle , \end{aligned} \tag{3.31}$$

since then it would not correspond to an injective function. We need then $G$ to act as the identity on a space that is wide enough to encompass what such states in (3.31) look like to *future* copies of $G$, but which also remains orthogonal to the states of the form in (3.31) themselves. Such a space is found in the work from Yao, with the orthogonality with

respect to the space spanned by the outputs in (3.31) being derived using the Bernstein-Vazirani conditions that we discussed in Section 3.2.1. At a more abstract level, the orthogonality corresponds to the fact that the inner product between the evolution under $U$ of different standard basis states in $\mathcal{H}_N$ must be 0. Since neither this general principle nor the Bernstein-Vazirani conditions have a sense of directionality, the reasoning can also be extended to the space corresponding to what the entry of $G$ corresponding to position $i-1$ could see if we first move the head from position $i$. This justifies the definition tweak that we previously described to Yao's version of the gate $G$ in order to make all of its applications in a step of the simulation commutative.

Finally, we note that the work in [119] also briefly considers the natural question of finding an inverse transformation: generating a quantum Turing machine computation equivalent to a quantum circuit computation. It observes that there are two parts to the work involved in such a simulation. The first part consists of simulating elementary quantum gates with a quantum Turing machine computation, for which one can make usage of the ideas discussed in Section 6 of [23]. The second part corresponds to deterministically following the circuit diagram. This will determine the overall complexity of the construction, and will be reliant on the specific choice of encoding for the circuit diagram. For each choice of encoding, optimizing this step will then correspond to an exercise in the design of reversible deterministic Turing machine computations.

### 3.3.8 Sensitivity to model choice

As previously mentioned in Section 3.2.3, it is reasonable to consider modifications to quantum Turing machines and see whether our construction and proof extends to the simulation of these modified models.

It is easy to see that if one allows stationary tape movements, none of the general facts about causality that we use in of our simulation to apply Theorem 1 are violated, and the proof of correctness goes through without issues, with the only change being in the structure of $G$ that we get when we read off its entries from $\delta$.

Even further, if we have a 2-dimensional tape, our results on unitary causal systems can be applied as well. We provide more detail below of the extension of our construction to 2-dimensional tapes with a possibly stationary head. This level of specific detail is not provided as an endorsement of that particular modification, but rather as an explicit step-by-step example illustrative of the straightforward extension of our work to other models. The reasoning there can be extended to many other hypothetical situations (like say a

head that reads and writes to two adjacent cells), which can also then be handled within the causality framework involved in our proof.

However, a case where our proof here cannot be trivially extended is the 2-tape case, since as far as one of the heads is concerned, the other head can be anywhere on the other tape, which conflicts with the definition of causality that is involved in our construction. However, it is possible with some work to get around this obstacle, as will be discussed further in Chapter 4.

### Extension to 2-dimensional tapes

A quantum Turing machine on a 2-dimensional tape with a possibly stationary head corresponds to a transition function of the form

$$\delta : Q \times \Gamma \to \mathbb{C}^{Q \times \Gamma \times \{-1,0,+1\} \times \{-1,0,+1\}}. \tag{3.32}$$

Now there is a second decision to make regarding the movement of the head, since there is one decision for each of the axes of movement. More in detail, for $a \in \Gamma$ and $p \in Q$, $\delta(p, a)$ represents again the behavior of the tape head when it is in the state $p$ and over a cell containing the symbol $a$. If $\delta(p, a)[(q, b, D_1, D_2)] = \alpha$ for $(q, b, D_1, D_2) \in Q \times \Gamma \times \{-1, +1\} \times \{-1, +1\}$, we have that in such a situation $\alpha$ is the amplitude for the machine to write the symbol $b$ to the tape and transition into state $q$, while its coordinate along the first axis of movement changes by $D_1$ and its coordinate along the second axis of movement changes by $D_2$.

The configuration space is now spanned by all possible tuples of the form $(i, j, q, T)$, where $i, j \in \mathbb{Z}$, $q \in Q$, and $T : \mathbb{Z} \times \mathbb{Z} \to \Gamma$ is a function that assigns the blank symbol to all but a finite number of elements of $\mathbb{Z} \times \mathbb{Z}$. The finite-dimensional equivalent of the machine used for the purposes of our simulation will now run on a torus-shaped tape. The cells in this tape will then be indexed by the elements of $\mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}$, for $N = 2t + 1$. The configuration space for the torus-tape machine will meanwhile be spanned by all tuples of the form $(i, j, q, T)$, where $i, j \in \mathbb{Z}$, $q \in Q$, and $T : \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z} \to \Gamma$.

As for the registers in the simulation, there will be a register tuple $(\mathsf{S}_{i,j}, \mathsf{T}_{i,j})$ for each tape position $(i, j) \in \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}$, with a similar meaning as in the standard tape case. The global state space for these registers will be denoted as $\mathcal{K}_N$, and there will be again a unitary $V \in \mathrm{U}(\mathcal{K}_N)$ that acts on them as derived from the QTM transition function $\delta$. This unitary $V$ will again act non-trivially only on the subspace of $\mathcal{K}_N$ spanned by standard basis states where exactly one of the $\mathsf{S}_{i,j}$ registers has a nonzero value, and that value is

positive. That subspace will remain invariant under the action of $V$, and it suffices again to implement a circuit that agrees with $V$ on it.

The operator $F$ is exactly the same as in the standard case, and so is the product of operations $W$ described in Equation (3.17) and Equation (3.19) that is applied in the circuit in order to simulate one step of the quantum Turing machine. The applications of a $F$ conjugated by $V$ do again commute with each other since the applications of $F$ commute with each other, and it is proved through Theorem 1 that they can be implemented through a 9-local gate $G$. More in detail, there is one application of Theorem 1 for each position $(i, j) \in \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}$. In this application, $\Pi$ projects again into the subspace for $\mathcal{K}_N$ where exactly one of the $\mathsf{S}_{i,j}$ registers contains a non-zero value (either positive or negative). $\mathcal{X}$ corresponds again to the state space $\mathcal{S}_{i,j}$ for register $\mathsf{X} = \mathsf{S}_{i,j}$, while $\mathcal{Y}$ corresponds to the state space for its neighbours (including diagonal ones), and $\mathcal{Z}$ corresponds to the state space for the rest of registers, bundled in a register $\mathsf{Z}$. Graphically, we can represent this by saying that the state space $\mathcal{Y}$ corresponds to the following range of registers:

$$\mathsf{Y} = \begin{pmatrix} (\mathsf{S}_{i-1,j-1}, \mathsf{T}_{i-1,j-1}) & (\mathsf{S}_{i,j-1}, \mathsf{T}_{i,j-1}) & (\mathsf{S}_{i+1,j-1}, \mathsf{T}_{i+1,j-1}) \\ (\mathsf{S}_{i-1,j}, \mathsf{T}_{i-1,j}) & & (\mathsf{S}_{i+1,j}, \mathsf{T}_{i+1,j}) \\ (\mathsf{S}_{i-1,j+1}, \mathsf{T}_{i-1,j+1}) & (\mathsf{S}_{i,j+1}, \mathsf{T}_{i,j+1}) & (\mathsf{S}_{i+1,j+1}, \mathsf{T}_{i+1,j+1}) \end{pmatrix}. \tag{3.33}$$

$\mathsf{Y} \to \mathsf{X}$ causality follows again from the one-cell-at-a-time movement of the head along each of the two axes of movement, and $\Pi$ can be decomposed into two terms corresponding to the case where the one head is in $(\mathsf{X}, \mathsf{Y})$ and the case where it is in $\mathsf{Z}$, respectively. After consequently applying Theorem 1, the copy of $G$ corresponding to each value of $i$ and $j$ is then being applied to the $3 \times 3$ grid of registers corresponding to $\mathsf{X}$ and $\mathsf{Y}$. Since there are $O(t^2)$ possible choices of $(i, j)$, this leads to a $O(t^3)$ total size for the circuit, while the depth remains $O(t)$, since each application of $G$ overlaps only with a constant number of the other applications.

As for the specific action of $G$, it can again be read off from the fact that

$$\Pi(G \otimes \mathcal{I}_{\mathcal{Z}})\Pi = \Pi V^*((F \otimes \mathcal{I}) \otimes \mathcal{I}_{\mathcal{Y}\otimes\mathcal{Z}})V\Pi. \tag{3.34}$$

One observes then that the action of $G$ will be non-trivial only in the case where there is exactly one non-zero value in the $S_{i',j'}$ registers included in $\mathsf{X}$ or $\mathsf{Y}$, and furthermore, the value is negative when it corresponds to $\mathsf{S}_{i,j}$, and positive otherwise.

In $k$-dimensional settings, the construction here would again go ahead, with the main difference that $\mathsf{Y}$ would now contain $3^k - 1$ register tuples, and $G$ would consequently be $3^k$-local.

Yao's original method can also be extended to a 2-dimensional tape setting. However, the polynomials in the entries of $\delta$ (and their conjugates) that one needs to considerate explicitly in order to extend the construction and its proof do increase substantially in their complexity, and would so even further if one was to increase the dimension $k$ of the tape.

## 3.4 Equivalence between unitarity and isometricity for QTM evolution operators

As shown in [22, 23], any candidate transition function $\delta$ for a quantum Turing machine will induce a global unitary evolution in configuration space if and only if it induces a global isometric evolution. We will use our introduction of a looped-tape quantum Turing machine model in order to give a more abstract and computationally simpler proof for this statement.

### 3.4.1 Setting

For an arbitrary finite set of states $Q$ and alphabet $\Gamma$, we consider a QTM transition function candidate $\delta : Q \times \Sigma \to \mathbb{C}^{Q\times\Gamma\times\{-1,1\}}$ such that $\|\delta(q,\sigma)\| = 1$ for all $q \in Q, \sigma \in \Gamma$. This function $\delta$ induces a configuration space evolution $U_\delta \in \mathrm{L}(\mathcal{H})$, with $\mathcal{H}$ and $U_\delta$ defined as in Section 3.2.1. We will also similarly refer to $U_{\delta,N} \in \mathrm{L}(\mathcal{H}_N)$ as the evolution operator associated with using $\delta$ in a looped-tape QTM with $N$ cells.

In order to transform between the looped-tape QTM setting and the standard QTM setting, we define for each value of $M \in \mathbb{Z}/N\mathbb{Z}$ the linear operator $V_{N,M} : \mathcal{H}_N \to \mathcal{H}$, which embeds the finite tape into the infinite one by aligning the 0 position, and cutting the loop between position $M$ and position $M+1$. More rigurously, this operator is of the form

$$\sum_{(p,i,T)} |p, i, T_M\rangle \langle p, i, T| , \tag{3.35}$$

where the sum iterates over all standard basis states for $\mathcal{H}_N$ (i.e. all classical configurations), and

$$T_M(i) = \begin{cases} T(i), \text{ if } 0 \leq i \leq M \\ T(i+N), \text{ if } M-(N-1) \leq i \leq -1 \\ 0 \text{ (i.e. the blank symbol) otherwise} \end{cases} \tag{3.36}$$

For convenience, we will refer to $V_{N,N-1}$ (which simply assigns cell $i$ in the looped tape to cell $i$ in the infinite tape) as $V_N$.

### 3.4.2 Result and proof

The statement that we prove here is the following:

**Theorem 2.** *The operator $U_\delta \in \mathrm{L}(\mathcal{H})$ induced by a transition function candidate $\delta$ is a unitary if and only if it is an isometry.*

This statement is non-trivial since $\mathrm{L}(\mathcal{H})$ is an infinite-dimensional space. Note however that $\forall N$, $U_{\delta,N}$ is an isometry if an only if it is unitary, since it operates on a finite-dimensional Hilbert space. The statement in Theorem 2 follows then from the following statements, since the right-hand-side of the if and only conditions are equivalent, which establishes the equivalence of the left-hand-side conditions:

**Lemma 3.** *The operator $U_\delta$ induced by a transition function candidate $\delta$ is an isometry if and only if $U_{\delta,N}$ is an isometry for all values of $N \geq 5$.*

**Lemma 4.** *The operator $U_\delta$ induced by a transition function candidate $\delta$ is unitary if and only if $U_{\delta,N}$ is unitary for all values of $N \geq 5$.*

The proofs of these lemmas follow:

*Proof of Lemma 3.* $U_\delta$ will be an isometry if and only if for any two distinct elements $x_1$ and $x_2$ of the standard basis for $\mathcal{H}$, it holds that $\langle U_\delta x_1, U_\delta x_2 \rangle = 0$. Similarly, $U_{\delta,N}$ will be an isometry if and only if for any two distinct elements $y_1$ and $y_2$ of the standard basis for $\mathcal{H}_N$, it holds that $\langle U_{\delta,N} y_1, U_{\delta,N} y_2 \rangle = 0$.

To prove the $\Leftarrow$ direction of the lemma, observe that since $U_\delta$ moves the head one step at a time, there is a value of $N' \in \mathbb{N}$ such that $\forall N \geq N'$,

$$\langle U_\delta x_1, U_\delta x_2 \rangle = \langle V_N^* U_\delta x_1, V_N^* U_\delta x_2 \rangle = \langle U_{\delta,N} V_N^* x_1, U_{\delta,N} V_N^* x_2 \rangle . \tag{3.37}$$

Since $V_N^* x_1$ and $V_N^* x_2$ are members of the standard basis for $\mathcal{H}_N$, the desired statement follows.

To prove the $\Rightarrow$ direction of the lemma, consider two different states $y_1$ and $y_2$ of the standard basis for $\mathcal{H}_N$. If $N \geq 5$, then either the head position is the same for $y_1$ and $y_2$,

or the sets of possible head positions corresponding to $U_{\delta,N}y_1$ and $U_{\delta,N}y_2$ can overlap in at most one cell of the tape. Therefore, there exists a value $M \in \mathbb{Z}$ such that

$$\langle U_{\delta,N}y_1, U_{\delta,N}y_2\rangle = \langle V_{N,M}U_{\delta,N}y_1, V_{N,M}U_{\delta,N}y_2\rangle = \langle U_\delta V_{N,M}y_1, U_\delta V_{N,M}y_2\rangle\,. \tag{3.38}$$

(i.e. one can pick a value of $M$ such that the mapping $V_{N,M}$ does not cut-off the part of the looped tape through which $U_{\delta,N}y_1$ and $U_{\delta,N}y_2$ meet each other when mapping to a conventional two-way infinite Turing machine tape). The desired statement then follows.

□

*Proof of Lemma 4.* The lemma follows from Lemma 3 together with the following claim: if $U_{\delta,N}$ is unitary for all $N \geq 5$, then every standard basis element in $\mathcal{H}$ has a pre-image in $U_\delta$.

We will now prove that claim. To do so, we consider an arbitrary element $x$ in the standard basis for $\mathcal{H}$. For $N$ large enough, and any value of $M \in \{0,\ldots,N-1\}$, we have that $V^*_{N,M}x \neq 0$, and therefore $\exists y \in \mathcal{H}_N$, $\|y\| = 1$, such that $y = U^{-1}_{\delta,N}V^*_{N,M}x$. Note that since $U^{-1}_{\delta,N} = U^*_{\delta,N}$, all head positions in $y$ must be adjacent in the looped tape to the head position for $V^*_{N,M}x$. Then, there will be at least one value of $M$ such that $V_{N,M}$ does not change the distance between the possible head positions for $y$ when applied, and it will hold then that $U_\delta(V_{N,M}y) = V_{N,M}U_{\delta,N}y = x$, so $x$ has a pre-image in $U_\delta$, as desired. □

### 3.4.3 Generalizations

The proof given here extends without issues to the case of quantum Turing machines with a multi-dimensional tape, as well as quantum Turing machines where the tape must remain stationary, since each of the individual steps in the reasoning will go through with only cosmetic modifications.

However, there are limits to the extent to which the statement we prove can be applied to families of automata-like systems. For a simple construction of an automata-like system whose global evolution is isometric but not unitary, one can consider a one-way infinite tape where the system's evolution transfers the state of a cell into the cell to its right, and assigns a blank state to the left-most cell. For a more complex example in a two-way infinite tape, one can find an example in Section 2 in [19] that uses the introduction of an entanglement pair in distant parts of the system.

It might be then of interest to further characterize the set of quantum cellular automata evolutions (giving e.g. sufficient conditions in terms of symmetries) for which a characterization like the one here can be conducted.

# Chapter 4

# Multi-tape quantum Turing machines

This chapter is organized as follows. In Section 4.1 we introduce multi-tape quantum Turing machines, while in Section 4.2 we examine the extension to that setting of our variant for the circuit simulation of quantum Turing machines. Then, we consider in Section 4.3 the additional presence of oracles.

This consideration of multi-tape quantum Turing machines represents a natural extension of the study of single-tape ones, as seen by the presence of work [81] that extends Yao's circuit simulation method to the multi-tape case. More generally, the naturalness of this extended study can also be observed in the presence of multi-tape machines in standard computing texts [5, 100] when discussing classical Turing machines. As we will further discuss in Section 4.2.4, the advantages of our variant here will be particularly notable in this multi-tape case in terms of simplification of the corresponding algebraic manipulations.

## 4.1 Setting

A multi-tape quantum Turing machine still has at any point a single state chosen from a set $Q$, but now reads and writes from $k$ two-way infinite tapes in each time step. These tapes correspond each to its own alphabet $\{\Gamma_0 \ldots, \Gamma_{k-1}\}$ . Formally, this corresponds to a transition function of the form

$$\delta : Q \times \Gamma_0 \times \ldots \times \Gamma_{k-1} \to \mathbb{C}^{Q \times \Gamma_0 \times \ldots \times \Gamma_{k-1} \times \{-1,1\}^k}. \tag{4.1}$$

This transition function has a similar meaning as that seen in Chapter 3 for a standard quantum Turing machine. More in detail, for

$$p \in Q, (a_0, \ldots, a_{k-1}) \in \Gamma_0 \times \ldots \times \Gamma_{k-1}, \tag{4.2}$$

$\delta(p, a_0, \ldots, a_{k-1})$ represents the behavior of the tape head when it is in the state $p$ and the $k$ heads are reading the symbols $a_0, \ldots a_{k-1}$ respectively. If

$$\delta(p, a_0, \ldots, a_{k-1})[(q, b_0, \ldots, b_{k-1}, D_0, \ldots, D_{k-1}] = \alpha \tag{4.3}$$

for

$$(q, b_0, \ldots, b_{k-1}, D_0, \ldots, D_{k-1}) \in Q \times \Gamma_0 \times \ldots \times \Gamma_{k-1} \times \{-1, +1\}^k, \tag{4.4}$$

this means that in such a situation $\alpha$ is the amplitude with which the state of the machine changes into $q$, and for all $i \in \{0, \ldots, k-1\}$ the head for tape $i$ writes the symbol $b_i$ to the tape, while adjusting its position in the tape as determined by $D_i$.

Similar to the single-tape case, we define a configuration space $\mathcal{H}$. Its standard basis corresponds now to the elements of the set

$$Q \times \mathbb{Z}^k \times \mathcal{F}_0 \times \ldots \times \mathcal{F}_{k-1}, \tag{4.5}$$

where $\mathcal{F}_j$ corresponds to the set of mappings of the form $\mathbb{Z} \to \Gamma_j$ such that only a finite numbers of inputs correspond to a non-blank value.

The transition function $\delta$ does again induce a transformation $U_\delta \in \mathrm{L}(\mathcal{H})$ by its action on the standard basis elements of $\mathcal{H}$, and this transformation must be unitary in order for $\delta$ to be a valid transition function. In order to define $U_\delta$ explicitly, we now re-introduce some notation regarding tape updates. In particular, given a tape assignment $T \in \mathcal{F}_j$ and $(i, a) \in \mathbb{Z} \times \Gamma_j$, $T_{i,a} \in \mathcal{F}_j$ will refer to a derived tape assignment. In this tape assignment, the symbol in position $i$ is assigned to be $a$, while all other positions take the same symbol as in assignment $T$.

Then, we have that $U_\delta$ performs the mapping where a standard basis element of $\mathcal{H}$ given by $\left|p, i, T^0, \ldots, T^{k-1}\right\rangle$ is mapped to

$$\sum_{q,a,D} \delta(p, T^0(i_0), \ldots, T^{k-1}(i_{k-1}))[q, a, D] \left| q, i + D, T^0_{i_0,a_0}, \ldots T^{k-1}_{i_{k-1},a_{k-1}} \right\rangle. \tag{4.6}$$

considering in the sum all choices of new state $q \in Q$, updated content under the heads $a \in \Gamma_0 \times \ldots \times \Gamma_{k-1}$, and direction of movement of the heads $D \in \{-1, 1\}^k$.

We again use looped tapes in order to more easily work with the spaces that arise during our work. For a tape length $N$, the corresponding configuration space will then be spanned by all tuples in the set

$$Q \times (\mathbb{Z}/N\mathbb{Z})^k \times \mathcal{F}_{N,0} \times \ldots \times \mathcal{F}_{N,k-1}, \tag{4.7}$$

where $\mathcal{F}_{N,j}$ is the set of all mappings of the form $\mathbb{Z}/N\mathbb{Z} \to \Gamma_j$. The transition function $\delta$ induces as in the standard a unitary evolution $U_{\delta,N}$ on this configuration space, and the fact that each head moves one step at a time means again that for $N \geq 5$, $U_{\delta,N}$ will be unitary if and only if $U_\delta$ was already unitary.

## 4.2 Extension to standard multi-tape quantum Turing machines of our variant for the simulation of quantum Turing machines

### 4.2.1 Setup for extension

Similar to the single tape case, our goal here is to describe (through a constructive and efficient procedure) a circuit that for a given multi-tape quantum Turing machine $M$ simulates for $t$ steps its computation on inputs of length $n \leq t$.

We assume that inputs are initially written on tape 0 starting on position 1, with their elements belonging to the corresponding alphabet $\Gamma_0$. Before the start of the computation, the head initially points to position 0 in all of the $k$ tapes. Also, we do again describe our circuit simulation in terms of simulating an equivalent looped-tape computation. There are $k$ such looped-tapes, with their length equal to $N = 2t + 1$, and their elements indexed for each tape by the elements of $\mathbb{Z}/N\mathbb{Z}$, which we will also write as $\{0, \ldots, N-1\}$ while implicitly assuming that index arithmetic is performed modulo $N$.

#### Circuit representation of the QTM components

Each tape cell there will be represented in our circuit by two registers. For the a cell in position $i \in \{0, \ldots, N-1\}$ of tape $j \in \{0, \ldots, k-1\}$, these registers will be labeled as $\mathsf{T}_i^j$ and $\mathsf{S}_i^j$. $\mathsf{T}_i^j$ stores the contents of the tape at that cell, while $\mathsf{S}_i^j$ contributes towards storing the current state and position of the head.

The state space for a register $\mathsf{T}_i^j$ is then such that the classical states for the register will be given by the members of $\Gamma_j$. These members will correspond then to the basis states for the state spaces $\mathcal{T}_i^j$ associated with each of the $\mathsf{T}_i^j$ registers, which within the same tape are all copies of the same state space $\mathcal{T}^j$. For example, in a machine with at least 3 tapes, the state spaces $\mathcal{T}_0^2, \mathcal{T}_1^2, \ldots, \mathcal{T}_{N-1}^2$ will all be a copy of the same space $\mathcal{T}^2$.

As for the state space for a register $\mathsf{S}_i^j$, the structure of the space $\mathcal{S}_i^j$ associated with the register will be a bit more complex. More in detail, for $i \in \{0, \ldots, N-1\}$ the state space for $\mathsf{S}_i^0$ will be spanned by standard basis elements labeled by the set $\{-|Q|, \ldots, 0, \ldots, |Q|\}$. The 0 label means that the head for tape 0 is not in position $i$. Otherwise, non-zero values represent the state of the head, and negative values represent additionally that the head has already moved. We denote the corresponding state space as $\mathcal{S}_0$. However, for $j \in \{1, \ldots, k-1\}$, we have that the classical state space for $\mathsf{S}_i^j$ corresponds to the simple set $\{0, 1\}$. 1 means that the head for tape $j$ is in position $i$ (no matter whether it has already been moved or not), while 0 means that it is *not* in cell $i$. The state space $\mathcal{S}_i^j$ for all values of $i \in \{0, \ldots, N-1\}$ and $j \in \{1, \ldots, k-1\}$ will then be a 2-dimensional state space, which we denote as $\mathcal{S}_b$.

In other words, the information about the current state and whether the heads have already moved will be kept in the registers associated with tape 0, while registers associated with the other tapes will only store information about the position of the corresponding heads. The $\mathsf{S}_i^j$ registers do then have for $j = 0$ a similar structure to that of the corresponding registers in the single-tape case, while for $j > 0$ they do have a different and simpler structure. This differs from the case of the $\mathsf{T}_i^j$ registers, where all registers have a similar structure to that of the corresponding registers in the single-tape case.

The global state space for all of the registers in the circuit will be denoted as

$$\mathcal{K}_N = \mathcal{S}_0^0 \otimes \mathcal{T}_0^0 \otimes \mathcal{S}_1^0 \otimes \mathcal{T}_1^0 \otimes \ldots \otimes \mathcal{S}_{N-1}^{k-1} \otimes \mathcal{T}_{N-1}^{k-1}. \tag{4.8}$$

We now translate the evolution of the QTM configuration space $\mathcal{H}_N$ to the evolution of the state space $\mathcal{K}_N$ for the circuit. We do this again by introducing an isometry $A$ that translates to the corresponding state in $\mathcal{K}_N$ the standard basis configurations of $\mathcal{H}_N$. The adjoint operator $A^*$ translates then to basis states of $\mathcal{H}_N$ those standard basis elements of $\mathcal{K}_N$ in which for every value of $j \in \{0, \ldots, k-1\}$, exactly one of the $\mathsf{S}_i^j$ registers has a non-zero value, and furthermore that value is positive in the case where $j = 0$ (i.e. it translates to $\mathcal{H}_N$ the basis states for $\mathcal{K}_N$ representing one head per tape, with the heads not marked as having been already moved). Then, the operator

$$V = AUA^* + (\mathcal{I} - AA^*), \tag{4.9}$$

where $U = U_{\delta,N}$, corresponds to the evolution of the state space $\mathcal{K}_N$ that we want to match with our circuit, as far as inputs in $\mathrm{Im}(A)$ are concerned. One can see that this translation is done in exactly the same way as in the single-tape case.

### Operators in simulation and their parallelization

A description of the operators involved in our variant for the circuit simulation of a multi-tape quantum Turing machine follow. The steps involved in their definition and analysis can be seen as generalizations of the steps taken for the single tape case in Chapter 3, with a varying level of complexity for those generalizations.

We first define an operator $F$, acting on $\mathcal{S} \otimes \mathcal{S}_b^{\otimes k-1}$ (i.e., on one head-storing register per tape). For standard basis states labeled by $|q, 1, \ldots, 1\rangle$, with $q \in Q$, this operator will transform the state into the one labeled by $|{-q}, 1, \ldots, 1\rangle$, and vice-versa. For all other computational basis states (those where at least one head is not in one of the registers that $F$ is acting upon), this operator will act as the identity. Note that as in the single-tape case, $F$ maps standard basis states to standard basis states, and it is equal to its own inverse. For any value of $i = (i_0, \ldots, i_{k-1}) \in \{0, \ldots, N-1\}^k$, we label as $F_i = F_{i_0,\ldots i_{k-1}}$ the operator that applies $F$ to registers $\mathsf{S}^0_{i_0}, \ldots, \mathsf{S}^{k-1}_{i_{k-1}}$, and acts as the identity on all other registers. The action of $F_i$ can be seen then as a unitary that can modify register $\mathsf{S}^0_{i_0}$ and is controlled by the state in registers $\{\mathsf{S}^1_{i_1}, \ldots, \mathsf{S}^{k-1}_{i_{k-1}}\}$.

The parallelization of the $F_i$ operators in our circuit is not obvious at first sight, since for different values of $i$ they can act on overlapping sets of registers, which makes their commutation properties non-trivial. In order to deal with this, we introduce the projection $\Pi \in \mathrm{Proj}(\mathcal{K}_N)$. This projection projects into the subspace spanned by standard basis states such that there is exactly one head per tape, no matter whether they have been marked as already moved or not (i.e., the subspace of $\mathcal{K}_N$ spanned by basis states such that for each value of $j \in \{0, \ldots, k-1\}$, there is exactly one value of $i$ such that $\mathsf{T}^j_i$ contains a non-zero value. We now claim that for any values of $i, i' \in \{0, \ldots, N-1\}^k$, it holds that

$$\Pi[F_i, F_{i'}]\Pi = 0. \tag{4.10}$$

and furthermore observe that the space $\mathrm{Im}(\Pi)$ is invariant under the action of $F_i$. This (partial) commutation claim follows from the fact that for any of the standard basis states $x$ that span $\mathrm{Im}(\Pi)$ there will be exactly one value $i'' \in \{0, \ldots, N-1\}^k$ such that $F_{i''}x \neq x$ (the value of $i''$ that corresponds to the position of the heads in $x$). Then, we combine this with the previously mentioned fact that for any value of $i'''$, $F_{i'''}$ is equal to its own inverse and corresponds to a permutation of the standard basis. Based on this, we can see that

for any standard basis state $x$ in $\mathrm{Im}(\Pi)$, $F_i F_{i'} x = F_{i'} F_i x$, which suffices to establish the relation in (4.10).

We have then that the product $F_i F_{i'}$ (or $F_{i'} F_i$) of two operators $F_i$ and $F_{i'}$ can be implemented in our circuit with two parallel applications of a gate $F$, provided that $F_i$ and $F_{i'}$ act on non-overlapping sets of registers and that all intermediate states during the execution of the circuit can be assured to belong to $\mathrm{Im}(\Pi)$ (which will be the case, as in the single-tape situation).

One can observe that this study of the parallelization properties of the $F_i$ operators was somewhat more involved than its equivalent in the single-tape case, and needed the introduction of the projection $\Pi$ at an earlier point of our analysis.

As our next step, we observe that in order to implement $V$, it will be equivalent to implement a circuit that agrees on $\mathrm{Im}(A) \subseteq \mathrm{Im}(\Pi)$ with the product of operators

$$W = F_{0,\ldots,0} F_{0,\ldots,0,1} \ldots F_{N-1,\ldots,N-1} (V^* F_{0,\ldots,0} V)(V^* F_{0,\ldots,0,1} V) \ldots (V^* F_{N-1,\ldots,N-1} V), \quad (4.11)$$

where the subscripts for $F$ iterate in lexicographical order over all subindex choices of one position per tape. This follows from considering the action of $W$ on standard basis states. More in detail, for every $F_0 F_1 \ldots F_{N-1}$ will map elements in $\mathrm{Im}(A)$ to elements in $\mathrm{Im}(A)^\perp$, on which $V^*$ will act as the identity. Furthermore, those elements of $\mathrm{Im}(A)^\perp$ will be in $\mathrm{Im}(\Pi)$, since $F$ does not modify the number of heads. Therefore, when an element $x \in \mathrm{Im}(A)$ is considered, it holds that

$$\begin{aligned} Wx &= (F_{0,\ldots,0} F_{0,\ldots,0,1} \ldots F_{N-1,\ldots,N-1})(F_{0,\ldots,0} F_{0,\ldots,0,1} \ldots F_{N-1,\ldots,N-1}) Vx & (4.12)\\ &= (F_{0,\ldots,0} F_{0,\ldots,0})(F_{0,\ldots,0,1} F_{0,\ldots,0,1}) \ldots (F_{N-1,\ldots,N-1} F_{N-1,\ldots,N-1}) Vx & (4.13)\\ &= Vx. & (4.14) \end{aligned}$$

We now observe that for any $i \in \{0, \ldots, N-1\}^k$

$$C_i = V^* F_i V \quad (4.15)$$

is the product of three unitary operators that each leave $\mathrm{Im}(\Pi)$ invariant, and therefore is itself a unitary operator that leaves $\mathrm{Im}(\Pi)$ invariant. We have then that for $i, i' \in \{0, \ldots, N-1\}^k$,

$$\Pi [C_i, C_{i'}] \Pi = V^* \Pi [F_i, F_{i'}] \Pi V = 0. \quad (4.16)$$

It follows then that the operators $C_i$ and $C_{i'}$ can be applied in any order when implementing the product $C_i C_{i'}$ (or $C_{i'} C_i$).

Furthermore, we know that if we have local implementations for the operators $C_i$ and $C_{i'}$ that do not act on overlapping sets of registers, they can be applied in parallel. It remains then to appeal to Theorem 1 to find local implementations for these operators. We will see in Section 4.2.2 how an approach that naively generalizes the setup for applying Theorem 1 in the single-tape case fails to establish this localizability. Then, in Section 4.2.3 we show how to modify this approach in order to make it go through in the multi-tape case.

### 4.2.2 Obstacle to naive proof for the extension

We are looking for a local implementation of the operator $C_i = V^* F_i V$, for an arbitrary value of $i = (i_0, \ldots, i_{k-1}) \in \{0, \ldots, N-1\}^k$. A generalization of our single-tape proof approach results then in an invocation of Theorem 1 where the registers $\mathsf{X}$, $\mathsf{Y}$, $\mathsf{Z}$, with matching state spaces $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$, are as follows:

- $\mathsf{X}$ corresponds to $\{(\mathsf{S}^0_{i_0}, \mathsf{T}^0_{i_0}), \ldots, (\mathsf{S}^{k-1}_{i_{k-1}}, \mathsf{T}^{k-1}_{i_{k-1}})\}$.
- $\mathsf{Y}$ corresponds to $\{(\mathsf{S}^0_{i_0-1}, \mathsf{T}^0_{i_0-1}), (\mathsf{S}^0_{i_0+1}, \mathsf{T}^0_{i_0+1}), \ldots, (\mathsf{S}^{k-1}_{i_{k-1}-1}, \mathsf{T}^{k-1}_{i_{k-1}-1}), (\mathsf{S}^{k-1}_{i_{k-1}+1}, \mathsf{T}^{k-1}_{i_{k-1}+1})\}$.
- $\mathsf{Z}$ corresponds to all other registers.

In other words, $\mathsf{X}$ corresponds to the $(\mathsf{S}^j_i, \mathsf{T}^j_i)$ register pairs that the $F$ operator is being applied to, while $\mathsf{Y}$ corresponds to the register pairs for the adjacent cells in each of the tapes, and $\mathsf{Z}$ corresponds to the rest of cells in the tapes.

The subspace within which our localization must work is the subspace $\Pi$ where there is one head per tape, as discussed and defined in Section 4.2.1. Informally, one can find a decomposition of $\Pi$ of the type required by the conditions in Theorem 1 by taking the decomposition for the single-tape case and raising it to the $k^{th}$ tensor product, while adjusting for the slightly different nature in the multi-tape case of the state spaces associated with the $\mathsf{S}^j_i$ and $\mathsf{T}^j_i$ registers.

More rigorously, we start with the tape $j = 0$, and consider projection families $\{\Delta^0_0, \Delta^0_1\}$ and $\{\Lambda^0_0, \Lambda^0_1\}$, defined along similar lines as $\{\Delta_0, \Delta_1\}$ and $\{\Lambda_0, \Lambda_1\}$ were in the single-tape case. For $h \in \{0, 1\}$, $\Delta^0_h$ projects then into the standard basis elements of $\mathcal{X} \otimes \mathcal{Y}$ where there is exactly one value of $i'_0 \in \{i_0 - 1, i_0, i_0 + 1\}$ (i.e. corresponding to the index range for $\mathsf{X}, \mathsf{Y}$) such that $(\mathsf{S}^0_{i'_0}, \mathsf{T}^0_{i'_0})$ represents a head. Meanwhile, $\Lambda^0_h$ has a similar meaning in terms of the basis elements of $\mathcal{Z}$ and the range of indices corresponding to $\mathsf{Z}$. We have then

that $\Delta_h^0$ projects into a copy of $(\mathcal{S} \otimes \mathcal{T}^0)^{\otimes 3}$, while $\Lambda_h^0$ projects into a copy of $(\mathcal{S} \otimes \mathcal{T}^0)^{\otimes N-3}$. For the tape $j > 0$, we similarly define projector families $\{\Delta_0^j, \Delta_1^j\}$ and $\{\Lambda_0^j, \Lambda_1^j\}$. The only difference with the $j = 0$ case is that for $h \in \{0, 1\}$, $\Delta_h^j$ and $\Lambda_h^j$ do now project into copies of $(\mathcal{S}_b \otimes \mathcal{T}^j)^{\otimes 3}$ and $(\mathcal{S}_b \otimes \mathcal{T}^j)^{\otimes N-3}$ respectively, as opposed to $(\mathcal{S} \otimes \mathcal{T}^0)^{\otimes 3}$ and $(\mathcal{S} \otimes \mathcal{T}^0)^{\otimes N-3}$. From this, we derive a global decomposition of $\Pi$ of the type required by Theorem 1. More explicitly, we can consider for example the case where $k = 2$, where the global decomposition of $\Pi$ is then given by

$$\sum_{h_0,h_1 \in \{0,1\}} (\Delta_{h_0}^0 \otimes \Delta_{h_1}^1) \otimes (\Lambda_{1-h_0}^0 \otimes \Lambda_{1-h_1}^1). \tag{4.17}$$

The family of operators $\{\Delta_1, \ldots, \Delta_n\}$ in the statement of Theorem 1 is then identified in the $k = 2$ case with

$$\{\Delta_0^0 \otimes \Delta_0^1, \Delta_0^0 \otimes \Delta_1^1, \Delta_1^0 \otimes \Delta_0^1, \Delta_1^0 \otimes \Delta_1^1\}, \tag{4.18}$$

while the family of operators $\{\Lambda_1, \ldots, \Lambda_n\}$ is identified with

$$\{\Lambda_1^0 \otimes \Lambda_1^1, \Lambda_1^0 \otimes \Lambda_0^1, \Lambda_0^0 \otimes \Lambda_1^1, \Lambda_0^0 \otimes \Lambda_0^1\}. \tag{4.19}$$

We do however find a problem when we try to complete this setup for the application of Theorem 1. The problem is that $V$ is in fact not $\mathsf{X} \to \mathsf{Y}$ causal within the one-head-per-tape subspace corresponding to $\mathrm{Im}(\Pi)$. To find an example of this, we can consider the two-tape case, and the situation where one head is with certainty in the region of tape 0 indexed by $\{i_0 - 1, i_0, i_0 + 1\}$, while the other head is with certainty not in the region of tape 1 indexed by positions $\{i_1 - 1, i_1, i_1 + 1\}$. Under those constraints, there are pairs of states $\rho$ and $\sigma$ that agree on their content in $(\mathsf{X}, \mathsf{Y})$ before applying $V$, but that due to disagreement about the tape content under the second head will not agree on the content of $\mathsf{X}$ after applying $V$.

It is possible in particular that for both $V\rho V^*$ and $V\sigma V^*$ there is a head in position $i_0 + 1$ of tape 0 with certainty, and that the head gets moved to position $i_0$ by $V$, but in different states when comparing $V\rho V^*$ and $V\sigma V^*$. This would be due to the head having read different symbols from tape 1 in the multi-tape QTM transition that $V$ represents. In other words, through the head in tape 1, what happens in $\mathsf{Z}$ affects the evolution of $\mathsf{X}$. It is necessary in light of this situation to make some tweaks in the proof approach in order to extent our construction to the multi-tape case, as will now be discussed in Section 4.2.3.

### 4.2.3 Making the extension work

The key step in order to successfully find a local implementation of $C_i$, therefore extending our simulation of quantum Turing machines to the multi-tape case, is to split the projection

$\Pi$ as a sum of two projections $\Pi_1 + \Pi_2$. In this sum, $\Pi_1$ projects into the subspace spanned by standard basis states where there is only exactly head per tape, and *all* heads are in a position corresponding to $\mathcal{X} \otimes \mathcal{Y}$ (i.e., the head for tape $j$ is in the range $\{i_j - 1, i_j, i_j + 1\}$ for all values of $j \in \{0, \ldots, k-1\}$. $\Pi_2$ corresponds to the complementary subspace of $\Pi_1$ relative to $\Pi$, spanned by the classical states where there is exactly one head per tape, and *at least* one head is *not* in a position corresponding to $\mathcal{X} \otimes \mathcal{Y}$ (i.e., the head for tape $j$ is *not* in the region indexed by $\{i_j - 1, i_j, i_j + 1\}$ for at least one value of $j \in \{0, \ldots, k-1\}$.

For example, in the two-tape case we have that now

$$\Pi_1 = (\Delta_1^0 \otimes \Delta_1^1) \otimes (\Lambda_0^0 \otimes \Lambda_0^1), \tag{4.20}$$

while

$$\Pi_2 = (\Delta_0^0 \otimes \Delta_0^1) \otimes (\Lambda_1^0 \otimes \Lambda_1^1) + (\Delta_0^0 \otimes \Delta_1^1) \otimes (\Lambda_1^0 \otimes \Lambda_0^1) + (\Delta_1^0 \otimes \Delta_0^1) \otimes (\Lambda_0^0 \otimes \Lambda_1^1). \tag{4.21}$$

We next apply Theorem 1 using the causality of $V$ relative to $\Pi_1$. The registers $\mathsf{X}, \mathsf{Y}, \mathsf{Z}$ and matchings state spaces $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ are the same as in Section 4.2.2. The reason why we can take this step now is that the type of counter-example presented in Section 4.2.2 cannot happen any more by definition of $\Pi_1$, and those are the only obstacles to having $\mathsf{X} \to \mathsf{Y}$ causality hold, given the one-step-at-a-time movement of the heads. This gives us then a gate $G$ that can be used to implement $C_i$ as far as inputs in $\mathrm{Im}(\Pi_1)$ are concerned.

We must now deal with the case where inputs to the local gate $G$ belong to $\mathrm{Im}(\Pi_2)$. Our argument to deal with that is that even if the statement of Theorem 1 does not guarantee it, the operator $G$ obtained from the previous application of Theorem 1 will work. That is to say, even if $G$ is derived in the context of using causality relative to $\Pi_1$, it does also give a correct output when it receives inputs belonging to $\mathrm{Im}(\Pi_2)$. In order to verify that this is the case, we examine the structure of the gate $G$ as determined by Equation (2.26) in Lemma 2. From this examination, one can see that $G \otimes \mathcal{I}_{\mathcal{Z}}$ will act as the identity on any state of the standard basis for $\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{Z}$ that is in

$$\ker((\Delta_1 + \ldots + \ldots + \Delta_n) \otimes \mathcal{I}_{\mathcal{Z}}) = (\ker(\Delta_1) \cap \ldots \cap \ker(\Delta_n)) \otimes \mathcal{Z}. \tag{4.22}$$

It is easy to see that the standard basis states that span $\mathrm{Im}(\Pi_2)$ do all belong to $(\ker(\Delta_1) \cap \ldots \cap \ker(\Delta_n)) \otimes \mathcal{Z}$, which corresponds for our invocation of Theorem 1 to $\ker(\Delta_1^0 \otimes \Delta_1^1) \otimes \mathcal{Z}$. This is due to the split between $\mathrm{Im}(\Pi_1)$ and $\mathrm{Im}(\Pi_2)$ in terms of the content of the $\mathsf{S}_i^j$ registers corresponding to $(\mathsf{X}, \mathsf{Y})$. In particular, this split means that such standard basis states are orthogonal not only to all the standard basis states that span $\mathrm{Im}(\Pi_1)$, but also to all the standard basis states that are obtained by starting with an

element of $\mathrm{Im}(\Pi_1)$ and then altering the content of register $\mathsf{Z}$. We have then by linearity that

$$\mathrm{Im}(\Pi_2) \subseteq (\ker(\Delta_1) \cap \ldots \cap \ker(\Delta_n)) \otimes \mathcal{Z} \tag{4.23}$$

and therefore (by (2.26)), that for all states $x \in \mathrm{Im}(\Pi_2)$, $(G \otimes \mathcal{I}_{\mathcal{Z}})x = x$. It remains to verify that this is the correct output of $C_i$ when applied to $x$. This follows from the fact that the $F_i$ will act as the identity. More in detail, we observe that for any standard basis state $x$ in $\mathrm{Im}(\Pi_2)$, if we compute $Vx$ and then measure the registers $\mathsf{S}^j_{i_j}$ for all values of $j \in \{0, 1, \ldots, k-1\}$, we will with probability 0 obtain the outcome where all of those registers contain a non-zero value (i.e. the outcome where all the registers represent a head). This follows from the fact that each of the heads in the machine move one step at a time, and for at least one tape $j$, the corresponding head is at least two positions away from position $i_j$ in $x$, as a pre-requisite of $x$ belonging to $\mathrm{Im}(\Pi_2)$. Therefore $F_i(Vx) = (Vx)$, and $C_i x = V^*(F_i V x) = V^* V x = x$.

It holds then that $C_i = V^* F_{i_0,\ldots,i_{k-1}} V$ agrees with $G \otimes \mathcal{I}_{\mathcal{Z}}$ on $\mathrm{Im}(\Pi_1) \cup \mathrm{Im}(\Pi_2)$, which implies by linearity that it also does so on $\mathrm{Im}(\Pi) = \mathrm{Im}(\Pi_1 + \Pi_2)$. We have then now successfully extended to the multi-tape case our argument for the ability to locally implement $C_i$ with a gate $G$.

Note that our work reveals that our reasoning in the single-tape case could have been modified so that in the invocation of Theorem 1 we have $\Pi = \Delta_1 \otimes \Lambda_0$. Then, the correctness of the action of $G$ for inputs within the subspace $\mathrm{Im}(\Delta_0 \otimes \Lambda_1)$ would be argued separately, as we do here for the multi-tape case.

### 4.2.4 Parallelism and complexity

In our complexity analysis, we take $k$ to be a constant, as well as the size of the state set and the size of the alphabet set for each of the tapes.

Regarding the complexity of the simulating circuit, we have that the total number of gates in the simulating circuit for a step of the multi-tape QTM is $O(t^k)$, counting all possible choices of $i$ for the $F_i$ and $C_i$ operators. The depth will be $O(t^{k-1})$, since any two of the $C_i = V^* F_{i_0,\ldots,i_{k-1}} V$ operators can be applied in parallel when they act on non-overlapping registers. This follows from the fact that this can be done as well with the $F_{i_0,\ldots,i_{k-1}}$ operators, as discussed earlier in Section 4.2.1. This means then that we can implement $\Theta(t)$ of the $C_i$ operators in parallel, reaching then the desired depth of $O(t^{k-1})$. After we multiply that by the factor of $t$ corresponding to simulating $t$ steps, we have then $O(t^{k+1})$ gates with $O(t^k)$ depth.

We will not address concerns about uniformity, alternate encodings and implementations with standard gate sets in depth in this section, since the discussion is not substantially affected when moving to the multi-tape case from the single-tape case. We refer the reader interested in those aspects to our previous analysis in Section 3.3.6 We note that asymptotic precision requirements for a standard gate set approximation do change, in that now the accuracy needed for each gate will be $O(\epsilon/t^{k+1})$, rather than $O(\epsilon/t^2)$. However, the qualitative result of then applying the Solovay-Kitaev theorem will not change, since such a precision is still possible to achieve with a size and depth blowout that is of polylogarithmic order in $t$ and $1/\epsilon$.

We can now verify that the depth of the circuit for simulating a single step of the simulation is $O(1)$ under the oblivious assumption that the tape positions at any point $t$ in the simulation are deterministic as a function of $t$. In particular, if the heads are known to be in position $i = (i_0, \ldots, i_{k-1})$ at time $t$, one can remove in the corresponding step of the simulation all gates except a number of them that is constant as a function of $t$ (but exponential as a function of $k$, which is a constant). The way to argue this is to remember that in the product

$$W = F_{0,\ldots,0}F_{0,\ldots,0,1}\ldots F_{N-1,\ldots,N-1}C_{0,\ldots,0}C_{0,\ldots,0,1}\ldots C_{N-1,\ldots,N-1}, \tag{4.24}$$

as far as inputs in $\mathrm{Im}(\Pi)$ are concerned we can reorder the product of $C$ operators as we prefer, and the same holds for the product of $F$ operators. This is due to the partial commutativity relations we established earlier. We can then write the product of $C$ operators so that it applies at the end the $3^k$ terms corresponding to $i' \in (\mathbb{Z}/N\mathbb{Z})^k$ such that

$$|i'_j - i_j| \leq 1, \forall j \in \{0, \ldots, k-1\}. \tag{4.25}$$

That is to say, we can move those terms to the left-most part of the product. Then, it is clear that all other $C$ operators will act as the identity, since for any such values of $i'$, we have that $F_{i'}$ will act as the identity in the product $C_{i'} = V^* F_{i'} V$, and $V$ and $V^*$ will then cancel each other. We can then remove all of those other $C$ operators. Similarly, we have that only the $5^k$ instances of $F$ corresponding to registers in the output of the non-removed $C$ operators might act non-trivially. Therefore, we can remove all other $F$ operators. This completes then our justification, and proves the corresponding claim in Section 3.3.6.

When comparing with previous knowledge in the literature, the key research to consider is that of Nishimura and Ozawa [81]. In that work, they extend the original single-tape construction from Yao [119] to the multi-tape setting. This results in a circuit with $O(t^{k+1})$ gates and $O(t^k)$ depth. The extension keeps the approach where part of the entries for the gate $G$ that is applied repeatedly are defined explicitly, while it is justified that a system of

equations used to find the rest of the entries must have a solution. In the multiparty case, the algebraic complexity of explicitly specifying a part of the entries before proceeding with the rest of the proof of correctness grows notably when compared with the single-tape case. In particular, the space where the gate $G$ acts as the identity has now a more complex structure, with the mathematical formulas in its description spanning more than half a page. At a high level, this complexity comes from a more involved structure for the subspace of what a copy of $G$ sees after a copy corresponding to a previous value of $i \in \{0, \ldots, N-1\}^k$ moves the head.

Something interesting is that in this multi-tape extension, the circuit is structured so that there is a single register to store the state that the head is in, as opposed to storing that in a register corresponding to one of the cells that the heads are in. When compared to our approach here, this is an interesting constant-order optimization to space storage that avoids duplicating space between the different $\mathsf{T}_i^0$ registers, which are zeroed when they do not correspond to a cell that the tape is currently reading from. Furthermore, it avoids the assignment of the state to one specific tape out of the $k$ tapes, which is asymmetric and therefore might be seen as a less elegant approach

However, this optimization makes it non-viable to perform the simple modifications to allow parallelism that we described in the case of Yao's original construction. This is because under the centralization of state storage we just described, each of the copies of $G$ needs to operate on the register that is storing the state, which means that only one of those gates can be applied at any given time, closing any avenues to parallelism.

If we modify the construction from [81] so that it instead stores the state together with the indicators about the position of the head, as the multi-tape version of our simulation variant does, we can then examine whether the tweaks for the parallelization of Yao's construction that we described in the single-tape case can be applied here to increase the degree of parallelism.

First, we need to consider the approach where we alter the definition of the space where Yao's gate $G$ acts as the identity. This works again in the multi-tape case, with the outcome that the output state after we move the heads using one copy of $G$ is not only left undisturbed by posterior copies of $G$ (in terms of lexicographical order for the corresponding index $i \in \{0, \ldots, N-1\}^k$), but also by previous copies of $G$ under that lexicographical order. The reasoning is substantially the same as in the single-tape case, relying again on the orthogonality of two subspaces of $\mathcal{X} \otimes \mathcal{Y}$. These two spaces are the space of what $G$ produces of its output after moving the heads, and the space of local states that $G$ could see as its input after the heads are moved by another copy of $G$. Since the head moves one step at a time, the overlap between two states that are respectively

in these two spaces will be equal to a linear combination where each term is a multiple of the global overlap $\langle Ux, Uy\rangle$ between the evolution under $U$ of two standard basis states $x, y \in \mathcal{H}_N$. All of those global overlaps must be 0, given the unitarity of $U$, leading to the orthogonality of the two subspaces.

It remains to discuss whether one can extend the cascading approach that in the case of Yao's construction allows one to achieve $O(1)$ amortized depth per simulation step without modifications to the gate $G$. This approach does not carry well to the multi-tape generalization of Yao's construction in [81]. The reason for this difference is that in the single-tape case, once that we have applied to the register pair $(\mathsf{S}_i, \mathsf{T}_i)$ representing the cell indexed by $i$ those instances of $G$ centered on positions $\{i-1, i, i+1\}$, we are ready (after the application of a 1-local gate $F$) to start with the second step of Yao's simulation as far as the processing of the register pair $(\mathsf{S}_i, \mathsf{T}_i)$ is concerned. This means that by the time we have finished the $O(t)$ gates in the first step of the simulation, we have also finished all but a $O(1)$ number of the number of gates for the second step of the simulation, leading to the amortized complexity advantage we described. The situation in the [81] case is more complex. This is because of the fact that there is one copy of the corresponding $G$ gate for each choice of central indices $i = (i_0, \ldots, i_{k-1}) \in \{0, \ldots, N-1\}^k$, and these gates are applied in lexicographical order. This means in particular that no cell in tapes $\{1, \ldots, k-1\}$ is done with its processing until we have processed all gates with a choice of index $i_0$ in the first tape belonging to $\{0, \ldots, N-2\}$. At that point in the simulation step, only $O(n^{k-1})$ gates remain to apply, which means that by the time we are done with the step, even if we cascade the next step there will still be at least $O(n^k)$ gates left to apply for it, with no amortized complexity gain.

In order to adapt the cascading approach to the parallelization of Yao's construction to the multi-tape case, one would then need to change the order in which the copies of $G$ can be applied, so that it is a better order for cascading purposes than the lexicographical order. To make the proof of correctness work, one would afterwards also need change the space on which $G$ is defined to act as the identity. But this is not any simpler than modifying those spaces to have full commutativity between the applications of $G$, as we described for the other parallelization approach, and would involve in fact more complicated algebra in the corresponding definition. We have then that the cascading approach to achieving parallelization of Yao's construction does now fall short of its single-tape advantage versus the definition-modification approach in terms of conceptual complexity.

## 4.3 Oracle quantum Turing machines

In this section, we discuss the adaptation of our multi-tape simulation variant to the case of oracle quantum Turing machines. After giving basic definitions in Section 4.3.1, we present an initial construction for this simulation in Section 4.3.2. This construction goes along similar lines to the standard multi-tape construction that we just described in Section 4.2, but now needs to deal with the more complex structure for the operator $U$ specifying the configuration space evolution. Then, we discuss in Section 4.3.3 how to improve on this construction by using a more standard model for oracle gates. Note that the material in this Section 4.3 represents a first step in the study of the circuit simulation of oracle quantum Turing machines, and the definitions and style might be less appealing than elsewhere in the thesis.

### 4.3.1 Definition

The standard model for oracle quantum Turing machines corresponds to work [18] from Bennett, Bernstein, Brassard, and Vazirani (BBBV), and takes them to be $k$-tape quantum Turing machines with a special oracle tape, which we will assume to be tape $k-1$. The alphabet $\Gamma_{k-1}$ for this tape is taken to be the set $\{0, 1, \square\}$, where $\square$ denotes the blank symbol (this breaks the convention generally followed in our work of denoting the blank symbol as 0, but will make the exposition in this section easier to follow). The oracle tape corresponds to a function $O$ from all binary strings to the binary set, which we denote as

$$O : \{0,1\}^* \rightarrow \{0,1\}. \tag{4.26}$$

There are also two special states, which we will refer to as $q_q$ and $q_a$. The way through which $f$ is computed is that if the machine is not in the state $q_q$, it evolves like a standard multi-tape quantum Turing machine. However, if the machine is in state $q_q$, and the oracle tape contains a binary substring $y = xb$, with $b \in \{0,1\}$, then the state becomes $q_a$ and the oracle tape now contains $y' = xb'$, where $b' = b \oplus O(x)$.

The configuration space $\mathcal{H}$ is the same one as for a standard $k$-tape quantum Turing machine. As for the evolution of the configuration space, we have now for oracle quantum Turing machines an operator $U_{\delta,O}$, depending on a oracle function $O$ and a transition function $\delta$. Its action on standard basis states will be as follows:

- If $p \neq q_q$, $\left|p, i, T^0, \ldots, T^{k-1}\right\rangle$ is mapped by $U_{\delta,O}$ to

$$\sum_{q,a,D} \delta(p, T^0(i_0), \ldots, T^{k-1}(i_{k-1}))[q, a, D] \left|q, i + D, T^0_{i_0,a_0}, \ldots, T^{k-1}_{i_{k-1},a_{k-1}}\right\rangle \tag{4.27}$$

where $T^j_{i,a}$ denotes the same modification of the tape assignment $T^j$ for tape $j$ as in the standard multi-tape case, with the character in position $i$ being replaced by $a$, and all other characters remaining the same.

- $\left|q_q, i, T^0, \ldots, T^{k-1}\right\rangle$ is mapped by $U_{\delta,O}$ to $\left|q_a, i, T^0, \ldots, T^{k-2}, f_O(T^{k-1})\right\rangle$, where $f_O$ is a transformation of the tape assignment $T^{k-1}$ that computes the function $O$ on tape $k-1$. More precisely, if the contents of tape $k$ are not of the form $\square^*\{0,1\}^*\{0,1\}\square^*$, where $^*$ denotes the Kleene star, $f_O$ acts as the identity. Otherwise, we can write the non-blank contents of tape $k$ as the concatenation of a string $x \in \{0,1\}^*$ and a bit $b$. Then, $f_O$ will rewrite the cell containing bit $b$ so that it now contains the binary value given by $b \oplus O(x)$.

Note that the definition in [18] does not explicitly state how the oracle QTM acts on oracle tape strings that have blanks interspersed with 0s and 1s, and instead constraints the transition function $\delta$ for a oracle quantum Turing machine to never give rise to such a situation. For completeness, we have specified the configuration space evolution to be the identity in those situations. However, we will make usage for our convenience of this fact, and only require in Section 4.3.3 from our circuit implementation of the oracle queries that it reproduces the correct behavior of the BBBV oracle QTM for the cases where its action was originally defined.

In our simulation, we will again consider an equivalent quantum Turing machine computation with looped tapes, with the same configuration space $\mathcal{H}_N$ as in the standard multi-tape case. This is allowed by the usual reasoning regarding the one-cell-at-a-time movement of the head. Note that it additionally implies now that only inputs of length $\leq N$ to the oracle (including the bit where the answer is written) are relevant in the corresponding simulation. The evolution $U = U_{\delta,O,N}$ of this configuration space will be extended from $U_{\delta,O}$ as usual for non-oracle tapes, having index arithmetic regarding the movements of the head now performed modulo $N$. In the oracle tape, we additionally need to specify how the looped-tape aspect interacts with the need to have only one segment of binary text in the string. We need this to match the behavior of a corresponding non-looped tape during the $t$ simulation steps that we want to perform. Therefore, for a classical state of the registers of our simulation, we look at the input to the oracle as the string associated with the content of the sequence of registers $(\mathsf{T}_{-t}, \mathsf{T}_{-t+1}, \ldots, \mathsf{T}_{t-1}, \mathsf{T}_t)$. This means a standard basis input state will be acted upon non-trivially by $U$ whenever the sequence of symbols corresponding to this sequence of registers is of the form $\square^*\{0,1\}^*\{0,1\}\square^*$.

### 4.3.2 A first circuit simulation

We use two registers $\mathsf{S}_i^j$ and $\mathsf{T}_i^j$ to represent cell $i$ in tape $j$ in the oracle Turing machine, with the same state spaces associated with these registers as in the standard multi-tape case described in Section 4.2. The global state space $\mathcal{K}_N$ associated with all the registers in our simulation will also be the same as in the standard multi-tape case.

We will denote again as $A$ the mapping that takes classical state configurations for $\mathcal{H}_N$ into standard basis elements of $\mathcal{K}_N$. In order to describe our circuit simulation for the oracle case, we must now introduce three new families of operators. In particular, for $q \in Q$, we will take $\Pi_q$ to be the projector that projects into the subspace of $\mathrm{Im}(A)$ where the corresponding QTM state is equal to $q$. Similarly, we will take $\Pi_{-q}$ to be the projector into the subspace of $\mathrm{Im}(A)$ where the corresponding QTM state is *not* equal to $q$. $\Pi_q$ will then map to themselves standard basis elements of $\mathcal{K}_N$ in which for every value of $j \in \{0, \ldots, k-1\}$, exactly one of the $\mathsf{S}_i^j$ registers has a non-zero value and the one for the tape $k = 0$ contains a value equal to $q$. All other standard basis states in $\mathcal{K}_N$ will be mapped to 0 by $\Pi_q$. A similar relation holds for $\Pi_{-q}$ , but now looking at the case where a positive value for the tape $k = 0$ is *not* equal to $q$. We also define the operator $\Pi_{q \to q'}$, which is parametrized by $q, q' \in Q$, and takes $\mathrm{Im}(\Pi_q)$ to $\mathrm{Im}(\Pi_{q'})$, while mapping everything else to 0. In particular, $\Pi_{q \to q'}$ will first project its input into $\mathrm{Im}(\Pi_q)$, and then change the value of the $\mathsf{S}_i^0$ register storing the state to be the value corresponding to $q'$.

Using the mapping $A$, we again extend the quantum Turing machine evolution operator to an operator on the simulation's registers by defining

$$V = AUA^* + (\mathcal{I} - AA^*), \tag{4.28}$$

Our goal now is to better understand the structure of $V$ so that we can see how to adapt the construction for the standard multi-tape case. We begin by writing

$$V = V_\delta + V_O + (\mathcal{I} - AA^*), \tag{4.29}$$

where $V_\delta$ is a unitary with support equal to $\mathrm{Im}(\Pi_{-q_q})$ and image equal to $\mathrm{Im}(\Pi_{-q_a})$, $V_O$ is a unitary with support equal to $\mathrm{Im}(\Pi_{q_q})$ and image equal to $\mathrm{Im}(\Pi_{q_a})$. There is no dependance of $V_\delta$ on $O$ or of $V_O$ on $\delta$. This block structure follows from the definition of $V$, together with the structure of $U = U_{\delta,O,N}$ that was described in Section 4.3.1.

We want to go from this decomposition of $V$ as a sum to a decomposition that expresses $V$ through multiplication, which will allow us then to have a circuit implementation of $V$ where one of the layers implements the part of the evolution corresponding to $\delta$, and the

other layer implements the part of the evolution corresponding to $O$. In order to get to such a multiplicative decomposition, we define operators $R_\delta$ and $R_O$, given by

$$R_\delta = V_\delta + \Pi_{q_q \to q_a} + \mathcal{I}_{\mathrm{Im}(A)^\perp} \tag{4.30}$$

$$R_O = V_O \Pi_{q_a \to q_q} + \mathcal{I}_{\mathrm{Im}(\Pi_{q_a})^\perp}. \tag{4.31}$$

It holds then that $R_O R_\delta = V$. In order to verify the correctness of this decomposition, one can simply appeal to linearity and check it for each element of the computational basis of $K_N$. As for the verification that $R_\delta$ and $R_O$ are both unitaries, one can observe that they are defined by a block decomposition, with each block being unitary itself, since $V_\delta$ and $V_O$ are unitary themselves. The reason why we must have the $\Pi_{q_q \to q_a}$ additive term in $R_\delta$ and the consequent $\Pi_{q_a \to q_q}$ multiplicative term in $R_O$ is to produce a block structure that maintains unitarity.

Our goal at this point is similar to the basic quantum Turing machine case. We want to describe a sequence of local gates and oracle gates such that for their product $P$, it holds that

$$A^* P A = A^* R_\delta R_O A = A^* V A \tag{4.32}$$

Since $R_\delta$ and $R_O$ both leave invariant the subspace $\mathrm{Im}(A)$, we can do so by coming up with two circuits that correspond to operators $P_\delta \in \mathrm{U}(\mathcal{K}_N)$ and $P_O \in \mathrm{U}(\mathcal{K}_N)$ such that $A^* P_\delta A = A^* R_\delta A$ and $A^* P_O A = A^* R_O A$.

For the circuit corresponding to $P_O$, we make usage of a model of oracle gates in which we assume that we have one gate (labeled as $G_O$) for which the corresponding operator acts on all of $\mathcal{K}_N$. The gate acts then acts on all the registers for the circuit, and it reproduces the effects of the oracle on the corresponding quantum Turing machine. That is to say, it corresponds to the operator $V_O$ in the decomposition appearing in Equation (4.29) that maps $\mathrm{Im}(\Pi_{q_q})$ to $\mathrm{Im}(\Pi_{q_a})$. We then complete $V_O$ to make it unitary, which results in the following definition:

$$G_O = V_O + \Pi_{q_a \to q_q} + \mathcal{I}_{\mathrm{Im}(\Pi_{q_a} + \Pi_{q_q})^\perp}. \tag{4.33}$$

The only standard basis states of $\mathcal{K}_N$ where $G_O$ acts non-trivially are therefore those on $\mathrm{Im}(A)$ where the only register $\mathsf{S}_i^0$ containing a non-zero value contains the value corresponding to $q_q$ or $q_a$. In the case where the value is $q_q$, the action of $G_O$ on those basis will update the value of the state so that it is equal to $q_a$, together with an update of the contents of the registers $\mathsf{T}_i^{k-1}$ associated with the last tape. On the case where the value of the state is equal to $q_a$, it is simply changed to $q_q$, with no additional updates. All other standard basis inputs will be mapped to themselves, with $G_O$ acting as the identity.

The circuit corresponding to $P_O$ can then be implemented with one copy of $G_O$, preceded and followed by an application of a 1-local gate $F_{q_q, q_a}$ to each register $\mathsf{S}_i^0$ corresponding to

the first tape. This gate $F_{q_q,q_a}$ acts on one copy of $\mathcal{S}_0$, mapping $|q_q\rangle$ to $|q_a\rangle$ and viceversa, while acting as the identity on all other standard basis states. The justification for correctness follows from considering every standard basis element of $\mathrm{Im}(A)$, and comparing the action of the circuit described here with that of the operator $R_O$ in Equation (4.31). Note that the entries in the matrix corresponding to $G_O$ depend exclusively on $O$, not $\delta$.

One might object that this gate $G_O$ can handle all queries to the oracle of length at most $N-1$, rather than queries of a specific length as it would be more standard. We will later address this matter in Section 4.3.3.

It suffices at this point for the purpose of our simulation to give a circuit with local gates such that for the induced operator $P_\delta$, it holds that $A^*P_\delta A = A^*R_\delta A$, as discussed before. In order to tackle this question, we use exactly the same approach as taken for the standard multi-tape case. The modifications that must be made are those corresponding to the fact that the operator $R_\delta$ in Equation (4.30) is not exactly the same operator as the one defined as $V$ in Equation (4.9).

In our simulation, we then still have two families of operators $F_i$ and $C_i$, indexed by all possible values of $i \in \{0,\ldots,N-1\}^k$. The $F_i$ operators are defined to be exactly the same as in the standard multi-tape case. The definition of $C_i$ needs to be modified to account for the fact that we now want to simulate $R_\delta$, with the following definition:

$$C_i = R_\delta^* F_i R_\delta \tag{4.34}$$

The circuit corresponding to $P_\delta$ is the same as that used to simulate $V$ in the standard multi-tape case, and corresponds then to the product

$$F_{0,\ldots,0}F_{0,\ldots,1}\ldots F_{N-1,\ldots,N-1}C_{0,\ldots,0}C_{0,\ldots,1}\ldots C_{N-1,\ldots,N-1}, \tag{4.35}$$

with the same justification by considering all standard basis elements in $\mathrm{Im}(A)$. Going forwards requires again introducing the projector $\Pi \in \mathrm{Proj}(\mathcal{K}_N)$, with the same definition as in the standard multi-tape case (i.e. projecting into the subspace of $\mathcal{K}_\mathcal{N}$ with one head per tape, no matter whether moved or not). It holds again that for any value of $i \in \{0,\ldots,N-1\}$, $C_i$ and $F_i$ leave $\mathrm{Im}(\Pi) \subseteq \mathrm{Im}(A)$ invariant, even after we changed the definition of $C_i$. It follows through the same reasoning as in the standard case that for any $i,j \in \{0,\ldots,N-1\}$,

$$\Pi[F_i,F_j]\Pi = 0 \tag{4.36}$$

and therefore

$$\Pi[C_i,C_j]\Pi = 0, \tag{4.37}$$

which gives us the commutativity relations needed for parallelization.

The final step is to apply Theorem 1 in order to achieve a local implementation of the operator $C_i$ for inputs in $\mathrm{Im}(\Pi)$. We follow the same approach as in Section 4.2, with the same assignment of registers to $\mathsf{X}$, $\mathsf{Y}$ and $\mathsf{Z}$, and the same split of $\Pi = \Pi_1 + \Pi_2$ with $\Pi_1$ corresponding to the case where all of the heads are in the registers corresponding to $\mathsf{X}$ and $\mathsf{Y}$). The only consideration that does then need to be double-checked is the one regarding $R_\delta$ being $\mathsf{X} \to \mathsf{Y}$ causal relative to $\Pi_1$. This is indeed true, since the action of $R_\delta$ will either write under the heads and move them one step following the rules in $\delta$, or simply leave the heads where they are and change the state to from $q_q$ to $q_a$, and differentiating between these two cases requires no information outside that contained in the registers associated with the initial position of the heads.

### 4.3.3 More complex circuit simulations with more standard oracle gate models

One can reasonably object to the circuit simulation in Section 4.3.2 by pointing out that while having oracle gates not be local is standard, our choice of oracle gate $G_O$ does not match some of the other aspects of standard oracle gate models. In particular, in standard oracle gate models there is one gate for each input length, as discussed for example in [113, 31]. We will now discuss the subject of achieving a circuit simulation with an oracle gate model where there is a separate gate for each input length.

In this discussion, we will take advantage of the flexibility provided by the standard definition of oracle QTMs in [18], and not consider the region of the configuration space spanned by standard basis elements where the content of the oracle tape contains several binary strings. We consider then an operator $A_{BBBV}$ that translates into the corresponding assignment to our registers those standard basis elements of the configuration space $\mathcal{H}_N$ where the content of the oracle tape is well-formed (i.e. of the form $\square^*\{0,1\}^*\square^*$). The image of this operator corresponds then to the subspace of concern regarding the correctness our simulation.

For each value of $l \in \{0,\ldots,N-1\}$ we consider then an oracle gate $G_{O,l}$, which communicates the value of $O$ for inputs of length $l$. Each copy of this gate will act on $l+1$ of the $\mathsf{T}_i^{k-1}$ registers, with the extra register corresponding to the one where the answer to the query is written. The input state space for $G_{O,l}$ is then $(\mathcal{T}_{k-1})^{\otimes(l+1)}$. A standard basis state $|y,b\rangle$ representing a binary sequence $y \in \{0,1\}^l$ and a bit $b \in \{0,1\}$ is mapped by $G_{O,l}$ to the standard basis state $|y, b \oplus O(y)\rangle$. On other standard basis inputs (i.e. those representing a string in $\Gamma_k^{l+1}$ that contains blanks), the action of $G_{O,l}$ might be taken to be the identity.

Our goal now is then to provide a circuit corresponding to an operator $P_O$ such that $A^*_{BBBV} P_O A_{BBBV} = A^*_{BBBV} R_O A_{BBBV}$, where $R_O$ is defined as in Equation (4.31). The part $P_\delta$ of the simulating circuit that corresponds to the transition function $\delta$ does not need to be modified from Section 4.3.2.

The main issue that we face now is that the content of tape $k-1$ can represent a binary query string of any length between 0 and $N-1$, so we must be able to select in the circuit corresponding to $P_O$ which of the oracle gates should be used for each of the possible inputs. Furthermore, we need to select the specific registers to which this gate should be applied. In line with our general approach to the simulation of quantum Turing machines, our goal is also to parallelize these choices as much as possible.

Our approach to dealing with these matters is to design classical (irreversible) circuits that each implement part of the corresponding logic, and then use standard constructions that transform these into reversible circuits suitable for integration in a quantum circuit. Those reversible circuits will each write their output through XOR operations on auxiliary inputs, and after using these outputs we must also be able to bring back the auxiliary inputs to their starting state, so that no interference patterns in the circuit are unduly altered.

### First reversible circuit

The first reversible circuit that we create has the purpose of determining where the endpoints of the query string are (and implicitly what its length is). The operator associated with this circuit will operate on all of the $\mathsf{T}_i^{k-1}$ registers associated with tape $k-1$, together with $(N^2+N)/2$ auxiliary qubits, initialized to the all-zero state. The state space associated with the input and the output of the circuit is then

$$(\mathcal{T}_{k-1})^{\otimes N} \otimes (\mathbb{C}^2)^{(N^2+N)/2}. \tag{4.38}$$

We define the operator by its action on the standard basis state $|x\rangle\,|y\rangle$ corresponding to a classical assignment $x \in \square^*\{0,1\}^*\square^*$ of length $N$ to the oracle tape values, together with a binary sequence $y$ of length $(N^2+N)/2$. This state will be mapped to a standard basis state $|x\rangle\,|y \oplus g(x)\rangle$, where $g(x)$ is a binary sequence indexed by all pairs $(p_1, p_2) \in \{-t, \ldots, t\}^2$ such that $p_1 \leq p_2$. $g(x)$ is such that the corresponding binary value is 1 only for the index value $(p_1, p_2)$ that represents the first and last non-blank position in the oracle tape. $g(x)$ acts then as a sequence of indicator variables representing the position in the oracle tape of the query string. If $x$ is the all-blank string, $g(x)$ is the all-zero sequence.

In order to obtain an implementation of this first circuit, it follows by standard results [16, 105] in reversible computing that it suffices to obtain an irreversible circuit that computes $g(x)$ given $x$. We now describe such a circuit with $O(\log t)$ depth.

The circuit starts by computing the integers $p_1$ and $p_2$ such that the non-blank section in the oracle tape is between positions $p_1$ and $p_2$, inclusive (if the query string is all-blank, we have that $p_1 = p_2 = t + 1$). We describe this computation process for $p_1$, with $p_2$ being calculated in a similar manner.

To do this, we first consider a gate for each two adjacent positions in $\{-t, \ldots, t\}$. Except in one corner case, these gates are such if the left position out of the two has a blank in its corresponding register of the oracle tape and the right position out of the two does not, we output the index corresponding to the right position, and otherwise we output $t + 1$. The corner case corresponds to the pair $(-t, -t+1)$, in which we case we first output the index corresponding to the left position in case that the corresponding register does not contain a blank, before proceeding with the previously described general logic. These gates can be implemented in two layers.

To finish computing $p_1$, we group the outputs of the previous step into adjacent pairs (if an output remains unmatched due to an odd number of outputs, it can be ignored and passed to the next step), and have a gate on each group. Given the two inputs to this gate, if one of them is not $t+1$ we output it, otherwise we output $t+1$ (if both inputs are not $t+1$ we output the smallest input, for completeness, but this case will not be relevant for our inputs of concern to $P_O$ in $\mathrm{Im}(A_{BBBV})$). These operators can all be implemented in a single layer. We then repeat this process recursively, until only one combined output remains. This computes $p_1$ as a number between $-t$ and $t + 1$. $p_2$ can be computed in a similar manner.

Now that we have $p_1$ and $p_2$, it remains to transform this into a binary sequence of indicator variables corresponding to $g(x)$. In order to do this, we first recursively use fan-out gates in order to make in $O(\log N) = O(\log t)$ depth $(N^2 + N)/2$ copies (i.e. one per entry in $g(x)$) for each of the $O(\log t)$ bits corresponding to $p_1$ and $p_2$, including the sign bit. We then consider $(N^2 + N)/2$ output wires, one for each of the entries in $g(x)$. Each output wire corresponding to an index tuple $(p'_1, p'_2)$ is initially set to 1, and we then have a sequence of $O(\log t)$ gates than can set it to 0. Each of the gates corresponds to a bit of the values $p_1$ and $p_2$ that we computed. The gate takes the copy of this bit that matches the current index in $g(x)$, and it compares it against the corresponding bit in $p'_1$ or $p'_2$, respectively (which is a fixed constant hard-wired into the definition of the gate). If the bits do not match, the corresponding entry of $g(x)$ is set to 0.

**Second reversible circuit**

The second reversible circuit that we create has the role of applying $O(t^2)$ oracle gates in parallel. There will be $(N^2+N)/2$ of these oracle gates, one for each of the possible tuples $(p_1, p_2)$ indicating a beginning and an end of the query string in the oracle tape

This circuit will act on all of the $\mathsf{T}_i^{k-1}$ registers corresponding to the content of the oracle tape, as well as on $(N^2+N)/2$ auxiliary qubits initialized to the all-zero state. The state space associated with the input and the output of the circuit is then

$$(\mathcal{T}_{k-1})^{\otimes N} \otimes (\mathbb{C}^2)^{(N^2+N)/2}. \tag{4.39}$$

We now discuss the behavior of this operator on inputs of the form $|x\rangle |y\rangle$, where $|x\rangle$ corresponds to a classical assignment $x$ to the oracle tape content of length $N$ and with form $\square^*\{0,1\}^*\square^*$, and $y$ is a binary sequence of length $(N^2+N)/2$. Such inputs will get mapped to $|x\rangle |y \oplus h(x)\rangle$, where $h(x)$ is a binary sequence indexed by all pairs $(p_1, p_2) \in \{-t, \ldots, t\}^2$ such that $p_1 \leq p_2$. To describe $h(x)$, we introduce the notation $x[p_1, p_2]$ to denote the substring of $x$ between positions $p_1$ and $p_2$, inclusive (if $p_1 > p_2$, $x[p_1, p_2]$ is the empty string, and $x[p_1, p_1]$ can also be written as $x[p_1]$). The element of $h(x)$ corresponding to the index pair $(p_1, p_2)$ will be equal to $O(x[p_1, p_2 - 1])$ whenever $x[p_1, p_2]$ is a binary string, with $O$ being the oracle function. Otherwise, that element of $h(x)$ will be equal to 0.

Regarding the implementation of this operator, we again begin by describing how to compute $h(x)$ with a circuit that includes irreversible classical gates. This circuit begins by computing $(N^2+N)/2$ copies of $x$ by recursively using classical fanout gates, with a total depth of order $O(\log N) = O(\log t)$. We again index these copies of $x$ by the set of all pairs $(p_1, p_2) \in \{-t, \ldots, t\}^2$ such that $p_1 \leq p_2$. We then act with an oracle gate on each of these copies of $x$, which takes $O(1)$ depth.

In particular, given the copy of $x$ corresponding to an index $(p_1, p_2)$, we will act with a copy of $G_{O,p_2-p_1}$ on the registers between position $p_1$ and position $p_2 - 1$, together with an auxiliary register with state space $\mathcal{T}_{k-1}$ that is initialized to the $|0\rangle$ state. Since the value in the last register is initialized to $|0\rangle$ before applying $G_{O,p_2-p_1}$, it will now contain $|O(x[p_1, p_2 - 1])\rangle$ whenever $x[p_1, p_2 - 1]$ is a binary string, and $|0\rangle$ otherwise. We must now adapt this to the definition of $h(x)$, which is conditional on $x[p_1, p_2]$ being a binary string, rather than $x[p_1, p_2 - 1]$ being so. This can be done by checking the value of $x[p_2]$. If it is a blank, we set the value in this auxiliary register to $|0\rangle$. Then, we output the value in the auxiliary register as the corresponding element of $h(x)$.

We then apply again standard constructions [16, 105] to make the circuit reversible. These techniques do embed each non-reversible gate into a larger reversible gate. Since we only have access to oracle gates in a black-box fashion, this might at first sight look problematic, but the fact that oracle gates are already reversible makes it not an issue. The techniques also require us to have access to the inverse of each resulting reversible gate, but since the oracle gates are involutory (i.e. equal to their own inverse), this is not an issue either.

### Third reversible circuit

The third reversible circuit has the role of translating the output of the previous steps into choices about whether to leave as constant or flip each binary digit in the oracle tape. Note that for any standard basis input to $P_O$ in our subspace of concern $\mathrm{Im}(A_{BBBV})$, at most one of the bits in the oracle tape will be flipped (the bit corresponding to a last position in the binary substring of the oracle tape). Making these choices involves access to the state for the oracle quantum Turing machine, so that we know whether an oracle query is being made in this step of the simulation. The state space for the input and output of this circuit is then

$$(\mathcal{S}_0)^{\otimes N} \otimes (\mathbb{C}^2)^{(N^2+N)/2} \otimes (\mathbb{C}^2)^{(N^2+N)/2} \otimes (\mathbb{C}^2)^N. \tag{4.40}$$

The first term correspond to the registers $\{\mathsf{S}^0_{-t}, \ldots, \mathsf{S}^0_t\}$ that store information about the state of the machine. The second term corresponds to the registers where the output was written in the first reversible circuit. The third term does similarly correspond to the registers where the output was written in the second reversible circuit. The fourth term corresponds to auxiliary qubits that are initialized to the all-zero state before we apply this third reversible circuit, and where we write the answers it computes.

As for the specific action of the reversible circuit, it will map a standard basis state $|s\rangle |w\rangle |y\rangle |z\rangle$ to $|s\rangle |w\rangle |y\rangle |z \oplus o(s,w,y)\rangle$, where $o(s,w,y)$ is a binary sequence of length $N$. This binary sequence is equal to the all-zero sequence if none of the symbols in $s$ is equal to $q_a$. This has the role of implementing the part of $R_O$ in (4.31) that corresponds to the control on the state being $q_q$ in $V_O$, as well as the term $\Pi_{q_a \to q_q}$. It remains then to define the behavior when at least one of the symbols in $s$ is equal to $q_a$. In this case, the element of the sequence $o(s,w,y)$ corresponding to position $p_2 \in \{-t, \ldots, t\}$ will be 1 if there is a value of $p_1 \in \{-t, \ldots, p_2\}$ such that the digits in $w$ and $y$ indexed by the pair $(p_1, p_2)$ are both equal to 1. Otherwise, the element of the sequence $o(s,w,y)$ corresponding to position $p_2$ will be equal to 0.

More specifically and relevant to the correctness of our simulation, if we consider a standard basis input to $P_O$ in $\mathrm{Im}(A_{BBBV})$, it will correspond to an assignment $|x\rangle$ to the $\mathsf{T}^{k-1}_i$

registers that store the content of the oracle tape, with $x$ of the form $\square^*\{0,1\}^*\square^*$. It will also correspond to an assignment $|s\rangle$ to the $\mathsf{S}_i^0$ registers that store the current state of the machine, with exactly one positive element in the sequence $s$, and all other elements equal to 0. Then, the input to this third reversible circuit will be $|s\rangle\,|g(x)\rangle\,|h(x)\rangle\,\left|0^N\right\rangle$, where $g(x)$ and $h(x)$ are the binary sequences computed in the first second reversible circuits. The output of the circuit will be $|s\rangle\,|g(x)\rangle\,|h(x)\rangle\,|o(s,g(x),h(x))\rangle = |s\rangle\,|g(x)\rangle\,|h(x)\rangle\,|o'(s,x)\rangle$. If the one positive value in $s$ is not $q_a$, then $o'(s,x) = 0^N$, and the circuit just acts as the identity. If the one positive value in $s$ is $q_a$, then we consider that at most one value of $g(x)$ will be non-zero – the one that matches the position of a binary sequence in the oracle tape. If there is such a value, it corresponds to a specific pair of indices $(p_1, p_2)$. Then, the value of $o'$ indexed by $p_2$ will be set to 0 or 1 depending on whether the entry of $h(x)$ also indexed by $(p_1, p_2)$ is equal to 0 or equal to 1 (i.e. depending on the value of $O(x[p_1, p_2 - 1])$, with $O$ the oracle function). All other values in $o'(s,x)$ will be equal to 0.

As for the implementation of this circuit, we approach it again by describing a classical circuit that computes $o(s,w,y)$ given the value of $s$, $w$ and $y$, and whose conversion to a reversible circuit of the type we desire is then straightforward through the usage of standard techniques.

The first step of the circuit is to extract a value $q \in \{-|Q|, \ldots |Q|\}$ from the string $s$ by combining adjacent pairs of symbols recursively, with a total depth of $O(\log N) = O(\log t)$. In this combination process, the circuit advances non-zero inputs to the next step when one input is zero while the other one is not, and it advances a zero when both inputs are zero. Other behaviors can be defined for completeness as returning the smaller of the inputs, but are irrelevant for the values of $s$ that may occur in our simulation, given the restriction to $\mathrm{Im}(A_{BBBV})$ as our subspace of concern. Similarly, non-positive values of $q$ will also never occur as the output of this step within the scope of our simulation.

The second step in the circuit is to take in one single layer of parallel gates the bitwise AND of $w$ and $y$. We label the corresponding binary sequence as $wy$. Then, for each index $p \in \{-t, \ldots t\}$, the corresponding element of $o(s,w,y)$ is taken to be the OR of the $O(N)$ values in $wy$ that are indexed by pairs of the form $(p_1, p_2)$, where $-t \leq p_1 \leq p_2 = p$. This can be computed again in $O(\log t)$ depth by recursively combining those values in $wy$.

Finally, we set all elements of $o(s,w,y)$ to 0 if the state $q$ computed in the first step of this third circuit does not correspond to the state $q_a$. This takes $O(1)$ depth for the comparison of $q$ with $q_a$, then $O(\log t)$ depth for creating $t$ copies of the comparison's result, and finally $O(1)$ depth for acting in parallel on each value of $o(s,w,y)$.

**Writing the answer and reversing auxiliary registers to their initial state**

We now write the output bits from the previous step into the oracle tape. In particular, for each value of $p_2 \in \{-t, \ldots, t\}$, the value of $h(x)$ indexed by $p_2 \in \{-t, \ldots, t\}$ is now combined with the value of the register $\mathsf{T}_{p_2}^{k-1}$ through a reversible 2-local operation that we label as $P - CNOT$. This operator has as the state space for its input and output the space $\mathcal{T}_{k-1} \otimes (\mathbb{C}^2)$. It will map $|a\rangle |b\rangle$ to $|a \oplus_g b\rangle |b\rangle$ for $a \in \{\square, 0, 1\}$, $b \in \{0, 1\}$, where $a \oplus_g b$ is equal to $\square$ if $a = \square$, and equal to $a \oplus b$ otherwise. This is equivalent to acting as the identity on standard basis inputs of the form $|\square\rangle |b\rangle$, for $b \in \{0, 1\}$, while mapping $|a\rangle |b\rangle$ to $|a \oplus b\rangle |b\rangle$ for $a, b \in \{0, 1\}$, similar to a standard $CNOT$ gate. It follows from our discussion regarding the output of the third reversible circuit for inputs in $\mathrm{Im}(A_{BBBV})$ that this accurately represents the action of an oracle transition on the contents of the oracle tape.

At this point, the only thing that remains to do is to reverse to the all-zero state those auxiliary inputs where the outputs for the first, second and third reversible circuits were written. We do this by applying these three circuits themselves again, starting with the third circuit. The correctness of this approach follows partially from the fact that all of these circuits are equal to their own inverse, which follows from the similar property of the bitwise $XOR$ operation. However, this correctness is non-trivial, since we have now potentially modified the value of the $\mathsf{T}_i^{k-1}$ registers, which are part of the input and output for the first and second reversible circuits. We must then verify that this reversion procedure works for each standard basis state of concern in the input to $P_O$.

As the setup for our verification of correctness, we consider an arbitrary standard basis input state to $P_O$ in $\mathrm{Im}(A_{BBBV})$. As described earlier, this corresponds to an assignment $|x\rangle$ with $x$ of the form $\square^*\{0, 1\}^*\square^*$ to the $\mathsf{T}_i^{k-1}$ registers that store the content of the oracle tape. Similarly, it corresponds to an assignment $|s\rangle$ to the $\mathsf{S}_i^0$ registers that store the current state of the machine, such that there is exactly one positive element in the sequence $s$, and all other elements are equal to 0. At the end of the third reversible circuit, the state of the registers in the circuit will then be equal to

$$|x\rangle |s\rangle |g(x)\rangle |h(x)\rangle |o(s, g(x), h(x))\rangle , \tag{4.41}$$

where the first two terms correspond to the $\mathsf{T}_i^{k-1}$ and $\mathsf{S}_i^0$ registers, and the last three terms correspond to the output registers for the three reversible circuits. After the application of the $P - CNOT$ gates, the content of these registers will be equal to

$$|x \oplus_g o(s, g(x), h(x))\rangle |s\rangle |g(x)\rangle |h(x)\rangle |o(s, g(x), h(x))\rangle , \tag{4.42}$$

For convenience, we will denote $x \oplus_g o(s, g(x), h(x))$ as $x'$. Note that in this expression, the operator $\oplus_g$ in the definition of $P - CNOT$ is being applied pair-wise to those elements in the sequences $x$ and $o(s, g(x), h(x))$ that correspond to a matching index.

Then, the correctness of our approach for reversing the output register of the third reversible circuit from $|o(s, g(x), h(x))\rangle$ back to $\left|0^N\right\rangle$ is not affected by the update from $|x\rangle$ to $|x'\rangle$ of the oracle tape content, since the $\mathsf{T}_i^{k-1}$ registers are not involved in the third reversible circuit.

As for the correctness of reverting $|h(x)\rangle$ back to $|0\rangle^{(N^2+N)/2}$ by then applying the second reversible circuit again, the situation is more complex, since the $\mathsf{T}_i^{k-1}$ registers are involved in this circuit. However, it holds that $h(x') = h(x)$, which makes it work. To see why $h(x') = h(x)$, consider that each entry in the output of $h$ corresponds to one index pair $(p_1, p_2)$ with $-t \leq p_1 \leq p_2 \leq t$. The set of those pairs $(p_1, p_2)$ such that $x[p_1, p_2]$ contains blanks is the same if we were to take substrings in $x'$ instead, so the corresponding entries of the output of $h$ are equal to zero in both $h(x)$ and $h(x')$. Otherwise, $x[p_1, p_2]$ contains only binary digits. Then, it holds that $x[p_1, p_2 - 1] = x'[p_1, p_2 - 1]$, and therefore $h(x) = h(x')$. The reason why $x[p_1, p_2 - 1] = x'[p_1, p_2 - 1]$ is that we know by the definition of $o(s, g(x), h(x))$ that the transformation $x \to x \oplus_g o(s, g(x), h(x))$ can only modify the last symbol in the one binary segment of $x$ (or no symbol, if $x$ was the all-blank string). The index for this last symbol can never be between $p_1$ and $p_2 - 1$ under the assumption that $x[p_1, p_2]$ is a binary string, since $x[p_2]$ is then a binary symbol.

The fact that $g(x) = g(x')$ does similarly make our approach work for reverting back to the all-zero state the auxiliary qubits where the answer to the first reversible circuit is written. In this case, it is an easy observation that $g(x) = g(x')$, since the endpoints of the non-blank segment in the oracle tape are the same in the states $|x\rangle$ and $|x'\rangle$. This follows from the fact that the transformation $x \to x \oplus_g o(s, g(x), h(x))$ does only modify non-blank values in $x$, by the definition of $\oplus_g$.

### Discussion

One can observe that our usage of classical low-depth circuits that are then transformed into reversible circuits is similar to the adaptation of output encodings that was described in Section 3.3.6, but with a significantly more complex chaining of transformations involved in the details of the construction here.

As for the complexity of the resulting oracle QTM simulation circuit, the circuit for the implementation of $P_O$ that we have described here in Section 4.3.3 has a depth of $O(\log t)$, with a width (i.e. space consumption) of $O(t^2)$. This space consumption takes into account

both the number of registers we explicitly use, and the number of classical irreversible local gates before we make each of the circuits reversible (since the space overhead introduced by the usage of standard techniques for making circuits reversible is proportional to that number of irreversible local gates).

We now put this together with the costs for the non-oracle part of simulating a step of the oracle QTM, corresponding to $P_\delta$ as implemented in Section 4.3.2. Then, the $O(t^{k-1})$ depth for $P_\delta$ absorbs the $O(\log t)$ depth for this step. The same happens for the $O(t^k)$ circuit size, since $k \geq 2$. However, the width grows from $O(t)$ to $O(t^2)$. The depth and size for a complete circuit simulating $t$ steps of an oracle QTM remain then at $O(t^k)$ and $O(t^{k+1})$ when compared with the construction in Section 4.3.2 that uses a less standard oracle gate model. However, the circuit width grows from $O(t)$ to $O(t^2)$.

If one wants to reduce the width of the circuit while possibly increasing its depth, it is possible to do so by slightly modifying the circuit that we have described here in Section 4.3.3 and exploiting tradeoffs between depth and width. In particular, we can exploit the fact that the value indexed by $p \in \{-t, \ldots, t\}$ in the output $o(s, g(x), h(x))$ of the third reversible circuit depends only on the values within $g(x)$ and $h(x)$ (i.e. the output of the first and second circuits) that are indexed by the $O(N) = O(t)$ pairs $(p_1, p_2)$ such that $-t \leq p_1 \leq p_2 = p$.

Using this property, one can replace the circuit that we have described here in Section 4.3.3 by a concatenation of $N = 2t + 1$ circuits, one for each value of $p \in \{-t, \ldots, t\}$. In the circuit corresponding to a particular of $p$, the first and second reversible circuits are modified so that they only compute outputs for the $O(t)$ pairs $(p_1, p_2)$ such that $-t \leq p_1 \leq p_2 = p$, with a reduction from $O(t^2)$ to $O(t)$ in the output size and circuit size. Then, the third reversible circuit uses the outputs from the first and second reversible steps to perform the computation of its output bit corresponding to position $p$, without any modifications to the construction we have described in Section 4.3.3. All of its other output bits are simply set to zero. Note that for any standard basis input state in $\mathrm{Im}(A_{BBBV})$, only the circuit that corresponds to the value of $p$ that matches the end position of the binary segment in the oracle tape will act non-trivially, since in all other cases all output bits of the third reversible circuit will be set to 0 (if the oracle tape is all blank, then all circuits will act trivially).

Through this tradeoff, the depth for the circuit implementing $P_O$ rises to $O(t \log t)$ and the width goes down to $O(t)$, with the size remaining constant at $O(t^2)$ after the split we have described. The result of exploiting the tradeoff is then that the width of the overall circuit for the simulation of $t$ steps of an oracle quantum Turing machine goes down from $O(t^2)$ to $O(t)$, while the depth goes up from $O(t^k)$ to $\max(O(t^k), O(t^2 \log t))$. The circuit

size remains unchanged at $O(t^{k+1})$.

### 4.3.4 Other models and further work

The standard oracle definition in [18] suffices for the purpose of studying standard quantum computational complexity classes, and either in its original form or very similar variants it is also widely used in the study of query complexity. It is however mentioned in [18] that there exist situations where one might want to consider alternative oracle definitions, such as example oracles in the context of computational learning theory [107]. One might also want to consider oracles that take advantage of the power of quantum computing to implement unitary operators. This is represented in the oracle model from [80], where the oracle $O$ encodes an arbitrary length-preserving unitary transformation from the space spanned by $\{0,1\}^*$ to itself, which can also be seen as an infinite collection $\{O_0, O_1, \ldots\}$ of unitaries, with the unitary $O_n$ acting on a space spanned by the elements of $\{0,1\}^n$.

It might be of interest to look into circuit simulation techniques for oracle quantum Turing machines where the machine makes usage of these alternative oracle models. Generally speaking, in order to make our construction in Section 4.3.3 work in alternate settings, one would need to be provided with inverses for each of the corresponding oracle gates, since we relied on each size-dependent oracle gate corresponding to the BBBV oracle being its own inverse.

It might also be of interest to consider the mixed-state model of [3], where quantum circuits make usage of subroutines that are equivalent to a probabilistic oracle (i.e. equivalent to a normalized function from $\{0,1\}^r$ to $\mathbb{R}^q_{\geq 0}$, for $r, q \in \mathbb{N}$). One could seek to model these oracles within the context of quantum Turing machines, which would require extending the formalism for quantum Turing machines so that it does not require that the global evolution of the system be unitary. A path to model these oracles could then start by considering some of the alternative classically-controlled quantum Turing machine models previously mentioned in Section 3.2.1.

One could also seek modifications to the oracle quantum Turing machine model in order to make the transformation of machine computations into circuit computations easier while retaining the same computational power. One possible idea is not entering and exiting the oracle through unique states $q_q$ and $q_a$, but rather having all QTM transitions induce a change in the oracle tape, which would avoid the explicit handling of the special state values $q_q$ into $q_a$ in the circuit simulation. One might also aim to constraint the query process by requiring oracle query strings to start in a specific position (the work in [80] does something similar by requiring the oracle tape's head to be in a specific position before

queries to the oracle occur, for subroutine nesting purposes). This could work well together with also modifying the oracle quantum Turing machine so that it always writes the answer to queries in the same position of the tape, no matter the length of the input query string. Finally, another potential change is to have only binary symbols in the oracle tape, with an escaping system that encodes the beginning and end markers for query strings.

# Part II

# Quantum prover-verifier interactions

# Chapter 5

# Quantum state exclusion

## 5.1 Setting

The task of quantum state exclusion corresponds to a setting where an agent Alice is given a quantum system. The state of this system is chosen at random between $n$ options $\{\rho_1, \ldots, \rho_n\}$, with corresponding non-zero probabilities $\{p_1, \ldots, p_n\}$. It is unknown to Alice which of the $\rho_i$ was chosen, but she does know the $\{\rho_i\}$ and $\{p_i\}$ values characterizing the corresponding distribution. Alice's goal in the state exclusion task is to be able to give an index $j$ such that the state was *not* prepared in the state $\rho_j$. When Alice can achieve this with probability 1, we will say that we have *perfect* state exclusion. This task of state exclusion has recently been studied at length in [12], and is at the heart of the celebrated PBR thought experiment [93], where [30] (the article from where we take the problem we solve in this Section) is credited as the original source for the concept. The concept of this task has also been used for proving results in the context of quantum communication complexity [90, 67, 49], as well as for designing quantum signature schemes [6].

Formalizing further this concept of state exclusion, [12] obtains the following semidefinite programming (SDP) formulation:

$$\begin{aligned} \text{minimize:} \quad & \sum_i p_i \langle M_i, \rho_i \rangle \\ \text{subject to:} \quad & \sum_i M_i = \mathcal{I} \\ & M_i \geq 0. \end{aligned} \tag{5.1}$$

where $M_i \geq 0$ means that $M_i$ is positive semi-definite. Being able to perform perfect state exclusion corresponds to the optimal value of this SDP being equal to 0. Similarly, any optimal solution to the semidefinite program corresponds to an optimal positive-operator valued measure (POVM) for state exclusion. Note that since we are only concerned with perfect state exclusion, we can just ignore the $p_i$ in the rest of this presentation, since whether the value of the SDP is 0 or not does not depend on them.

Perfect exclusion of quantum states is also a meaningful concept in the context of the foundations of quantum mechanics, in particular when considering the topic of quantum state compatibility. In that framework, one considers several quantum states $\{\rho_1, \ldots, \rho_n\}$ as different beliefs about the same system. Then, one can ask whether the outcome of a measurement on the system will disprove some of these beliefs, or they will all still be possible. In the latter case, we say that the states are *compatible* with each other. Different ways of formalizing this idea will lead to different definitions of quantum state compatibility. [30] proposes several formalizations, one of which corresponds to the impossibility of performing perfect state exclusion. Since this formalization is a generalization of previous work by Peierls [85], they refer to it as post-Peierls (PP) compatibility.

In more detail, the post-Peierls compatibility of several quantum states $\{\rho_1, \ldots, \rho_n\}$ (relative to a subset $S$ of all POVMs) means that for all measurements in $S$, there will be at least one outcome that can be obtained with non-zero probability for all of the possible states/beliefs $\{\rho_1, \ldots, \rho_n\}$. If we consider the negation of this definition, we obtain that this negation corresponds with the existence of a measurement in $S$ such that each outcome of the measurement excludes at least one of the quantum states, which corresponds to an agent being able to perform perfect state exclusion given a mixture of the quantum states $\{\rho_1, \ldots, \rho_n\}$ and access to measurements in $S$ [1]. When the set $S$ of allowed measurements corresponds to the set of all POVMs, the corresponding compatibility criteria is called PP-POVM compatibility. When $S$ is restricted to the set of projective measurements (or more precisely, the set of measurements defined by one-dimensional orthogonal projectors), [30] names the corresponding criteria as PP-ODOP compatibility.

One can consider the case where all of the $n$ states/beliefs $\{\rho_1, \ldots, \rho_n\}$ are known to belong to a particular subset $A$ of all quantum states, and ask whether for all such tuples of $n$ beliefs in $A$ the PP-ODOP and PP-POVM criteria will coincide with each other. When this happens, we will say that in that context PP-ODOP=PP-POVM. Note that this is equivalent to projections being optimal for perfect state exclusion within the context of input states in $A$.

[1] More formally, we can formalize PP compatibility through the equation $\forall\{\Pi_i\}\exists i\forall j \, \langle \Pi_i, \rho_j \rangle > 0$, which gives as its negation $\exists\{\Pi_i\}\forall i\exists j \, \langle \Pi_i, \rho_j \rangle = 0$

In [30], the authors identify a necessary and sufficient condition for PP-ODOP incompatibility of 3 pure states $\{a, b, c\}$ in 3 dimensions (i.e. they establish a condition for the states to be perfectly excludable via a projective measurement). This condition can be expressed in terms of the magnitudes of their inner products, given by $|\langle a, b\rangle|, |\langle a, c\rangle|, |\langle b, c\rangle|$, and which we will denote as $j_1$, $j_2$ and $j_3$, respectively. In particular, the condition obtained in [30] is that 3 pure states will be PP-ODOP incompatible whenever

$$j_1^2 + j_2^2 + j_3^2 + 2j_1j_2j_3 \leq 1. \tag{5.2}$$

We will refer to this formula as the **Caves-Fuchs-Schack inequality**, after the authors of [30]. Note that we have corrected in our presentation of this formula the original strict inequality sign that they use, following the indications to do so in [14, 101], and we have also merged the two conditions from the original presentation in [30] into one single condition.

When this result was introduced in [30], it was mentioned that the authors were not able to prove that PP-ODOP = PP-POVM in the context of 3 pure states in 3 dimensions, despite having numerical evidence that this is the case. The authors also present results establishing that this is the first open case – they cite previous work [73] showing that for two pure states in any dimension PP-ODOP = PP-POVM, and establish that for $k > 2$ pure states in 2 dimensions this will not necessarily be the case.

We will give now an analytical proof which answers the corresponding question, by showing that PP-ODOP = PP-POVM in the context of 3 pure states in 3 dimensions.

Our work can be seen as part of the line of work that studies POVMs in the context of low-dimensional systems of a fixed dimension. For example, [108] and [117] recently examined 2-dimensional POVMs in the contexts of nonlocal games and quantum state discrimination, respectively, while [120] looked into 4-dimensional POVMs in the context of imposing symmetry conditions.

We conclude with a discussion about different ways in which our work might be generalized. Of special interest here might be our discussion on the usage of Quadratically Constrained Quadratic Programming (QCQP) to model the $n$-dimensional variant of the question we solve. This is a type of mathematical optimization formalism that has seen a large number of applications in recent years, but only limited usage so far within the context of quantum information processing. To our knowledge, this is the first time that state exclusion of pure states through projections is expressed through a problem in a standard form of a mathematical optimization framework.

## 5.2 Main derivation

### 5.2.1 Restrictions that can be imposed without loss of generality on POVMs that achieve perfect exclusion

Our goal is to prove that for any set of 3 pure states in 3 dimensions that are perfectly excluded by a measurement (i.e. a POVM), they are also perfectly excluded by a projective measurement. Following Equation (5.1) and our analysis of it, we can identify the perfect exclusion of three pure states $a, b$ and $c$ with obtaining an optimal value of 0 in the following semidefinite program:

$$\begin{aligned} \text{minimize:} \quad & a^* M_1 a + b^* M_2 b + c^* M_3 c \\ \text{subject to:} \quad & \sum_i M_i = \mathcal{I} \\ & M_i \geq 0, \end{aligned} \tag{5.3}$$

Note that all the operators involved can be represented as $3 \times 3$ matrices. It is well-known in convex optimization that the solution to optimizing a linear function over a non-empty compact convex set in a finite dimensional Hilbert space can be assumed without loss of generality to be an extreme point of the set of feasible solutions[2] (note that in this case, that feasible set is the set of POVMs). Therefore, we can assume that at most one out of the $M_i$ has rank greater than 1. Otherwise, assume for the sake of contradiction that two of them (say $M_1$ and $M_2$) have rank at least 2, so there is a common vector $u$ in the images of $M_1$ and $M_2$. Then, for $\epsilon$ small enough both $\{M_1 + \epsilon uu^*, M_2 - \epsilon uu^*, M_3\}$ and $\{M_1 - \epsilon uu^*, M_2 + \epsilon uu^*, M_3\}$ are POVMs, which implies $\{M_1, M_2, M_3\}$ is not an extreme point of the feasible set.

Without loss of generality, we can permute indices so that the ranks of $M_1$, $M_2$, and $M_3$ are sorted in non-increasing order. Also, if $M_1$ has rank 3 it cannot exclude any quantum state, so it must be the case that its rank is at most 2. Note too that if the ranks are of the form $(1, 1, 1)$, it is not hard to see that the condition $M_1 + M_2 + M_3 = \mathcal{I}$ implies that $\{M_1, M_2, M_3\}$ form a projective measurement themselves. Similarly, in the case where the ranks are of the form $(2, 1, 0)$, $M_1 = \mathcal{I} - M_2$ implies that $M_1$ and $M_2$ form a projective measurement (this is because for the right hand side $\mathcal{I} - M_2$ to have rank 2, $M_2$ must

[2]This fact follows from applications of the Krein-Milman and Extreme Value theorems, which in their most general versions require in fact constraints less strong than the ones we have here.

have its non-zero eigenvalue equal to 1, which implies then the same for the eigenvalues of $\mathcal{I} - M_2 = M_1$).

We can then focus on the case where there is an optimal POVM $\{M_1, M_2, M_3\}$ with ranks of the form $(2, 1, 1)$. We also choose now without loss of generality to work in a basis such that $M_1$ is diagonal and it perfectly excludes $a = |0\rangle$.

We have then that $M_1$ will be determined by the choice of a real diagonal vector $(0, 1-x, 1-y)$, and $M_2$ and $M_3$ by a choice of complex vectors $v = (v_1, v_2, v_3)$ and $w = (w_1, w_2, w_3)$ such that $M_2 = vv^*$ and $M_3 = ww^*$. We claim now that we can assume $y = 0$, $v_1 \neq 0$, $w_1 \neq 0$, $v_2 \neq 0$, $w_2 \neq 0$, $v_3 = 0$, $w_3 = 0$. To see why, consider the following five observations:

1. The condition $M_1 + M_2 + M_3 = \mathcal{I}$ corresponds to the equations

$$v_1\overline{v_1} + w_1\overline{w_1} = 1 \tag{5.4}$$
$$v_2\overline{v_2} + w_2\overline{w_2} = x \tag{5.5}$$
$$v_3\overline{v_3} + w_3\overline{w_3} = y \tag{5.6}$$
$$v_1\overline{v_2} = -w_1\overline{w_2} \tag{5.7}$$
$$v_1\overline{v_3} = -w_1\overline{w_3} \tag{5.8}$$
$$v_2\overline{v_3} = -w_2\overline{w_3}. \tag{5.9}$$

2. We can assume $v_1 \neq 0$ and $w_1 \neq 0$, as otherwise $\{M_1, M_2, M_3\}$ can be trivially transformed into a projective measurement. To see this, suppose for example that $w_1 = 0$. Then, (5.4) implies that $|v_1| = 1$, (5.7) that $v_2 = 0$, and (5.8) that $v_3 = 0$. We have then that $M_2$ is diagonal, and its diagonal is equal to $(1, 0, 0)$. This implies that $M_3$ is diagonal as well, while $w_1 = 0$ implies that the first term in its diagonal is equal to 0, so we can group $M_1$ and $M_3$ into a single operator and obtain a projective measurement.

3. Suppose we had $v_2 \neq 0$ and $v_3 \neq 0$. Then, (5.9) implies $w_2 \neq 0$ and $w_3 \neq 0$. This means we can divide (5.7) by (5.8) and the conjugate of (5.9), and obtain that $\frac{v_1}{w_1} = \frac{v_2}{w_2} = \frac{v_3}{w_3}$. Let $\lambda$ be the value of these ratios. Then, each of equations (5.7)-(5.9) implies that $\lambda = 0$, which contradicts $v_1 \neq 0$.

4. We have then that either $v_2 = 0$ or $v_3 = 0$, and by symmetry we can assume without loss of generality that $v_3 = 0$. Then, $w_3 = 0$ as well, since otherwise (5.8) would imply $w_1 = 0$, which we know not to be the case. (5.6) implies then that $y = 0$.

5. If we were now to additionally impose that $v_2 = 0$, (5.7) would imply that $w_2 = 0$, which can only happen when $x = 0$, by (5.5). However, in the $x = 0$ case we have that $M_1$ is a projection on $|1\rangle$ and $|2\rangle$, and $M_2$ and $M_3$ can be merged into a projection on $|0\rangle$, so there trivially is an optimal projection for state exclusion, and the case is not of interest to us. We can assume then that $v_2 \neq 0$, and similarly that $w_2 \neq 0$.

We can introduce now a parameter $r$, which determines the distribution of the weight $x$ between $M_2$ and $M_3$, and let $|v_2|^2$ be equal to $x\frac{1}{r+1}$ ($r \in (0, \infty)$). We have then that (5.5) implies $|w_2|^2 = x\frac{r}{r+1}$, and that (5.7) and (5.4) imply then that $|v_1|^2 = \frac{r}{r+1}$, $|w_1|^2 = \frac{1}{r+1}$. The magnitudes of each element of $v$ and $w$ are then completely characterized by the values of $r$ and $x$.

As for the phases of the elements of $v$ and $w$, we can assume that $v_1, w_1 \in \mathbb{R}$ without affecting the values of $M_2$ and $M_3$. Then, if the phase of $v_2$ is given by $\theta$, (5.7) implies that the phase of $w_2$ is given by $\pi + \theta$.

We reach then our final form for what a POVM $\{M_1, M_2, M_3\}$ for perfect state exclusion of 3 pure states in 3 dimensions can be assumed to be without loss of generality. In matrix form, it is given by

$$M_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1-x & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.10}$$

$$M_2 = \frac{1}{r+1} \begin{pmatrix} r & e^{-i\theta}\sqrt{rx} & 0 \\ e^{i\theta}\sqrt{rx} & x & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.11}$$

$$M_3 = \frac{1}{r+1} \begin{pmatrix} 1 & -e^{-i\theta}\sqrt{rx} & 0 \\ -e^{i\theta}rx & rx & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.12}$$

where $0 < x < 1$, $r \in (0, \infty)$, $0 \leq \theta < 2\pi$.

### 5.2.2 Verification that any states perfectly excluded by our parametrized optimal POVM satisfy the Caves-Fuchs-Schack inequality

We look first at the structure of the states $b$ and $c$ perfectly excluded by $M_2$ and $M_3$, and obtain that it is enough to consider a one-parameter family for each of them. Let $b$ be

given by $(b_1, b_2, b_3)$, and $c$ by $(c_1, c_2, c_3)$. Then, our conclusion follows from the following five observations:

1. As usual, we can get rid of unphysical global phases, and assume $b_1$ is a real positive number. This is because multiplying $b$ by a phase will not affect the value of our semidefinite program (5.3), and it will not affect either the satisfaction of the Caves-Fuchs-Schack inequality.

2. It can be seen from (5.11) that the value of $b_2$ is completely determined by the value of $b_1$ by the constraint $M_2 b = 0$ (which is equivalent to $b^* M_2 b = 0$, since $M_2$ is positive semidefinite, making both conditions equivalent to $b \in \mathrm{Ker}(M_2)$). In particular, one obtains that $b_2 = -b_1 e^{i\theta}\sqrt{\frac{r}{x}}$.

3. The fact that $b$ has norm 1 (since it represents a pure state) allows us now to express the magnitude of $b_3$ as a function of $b_1$. In particular, the magnitude of $b_3$ is given by $\sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}$, while its phase, which we will denote by $\vartheta$, can take any value.

   Note that this implies an upper bound on $b_1^2$, given by $1/\left(1 + \frac{r}{x}\right)$.

4. A similar analysis applies to $c$, and we have that it can be parametrized by a real positive value $c_1$ such that $0 \leq c_1^2 \leq 1/\left(1 + \frac{1}{rx}\right)$, together with the phase $\gamma$ of $c_3$. In this case, the value of $c_2$ is given by $c_1 e^{i\theta}\sqrt{\frac{1}{rx}}$, and the magnitude of $c_3$ is given by $\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)}$.

5. We can assume now that the phases $\vartheta$ and $\gamma$ of $b_3$ and $c_3$ are selected in order to maximize the left hand side of the Caves-Fuchs-Schack inequality. The reason we can do this is because we are interested in proving that the Caves-Fuchs-Schack inequality holds, so this is a worst-case scenario in our situation.

   To do so, note that $j_1 = b_1$ and $j_2 = c_1$, so they do not depend on the phases of $b_3$ and $c_3$. Therefore, maximizing the left hand side of the Caves-Fuchs-Schack inequality will be equivalent to maximizing $j_3 = |b^* c|$. To do that, we compute first the value of $b^* c$, given by

$$e^{i(\gamma-\vartheta)}\sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)} + b_1 c_1 - \frac{1}{x} b_1 c_1 \tag{5.13}$$

The magnitude of this expression as a function of $\gamma$ and $\vartheta$ will be the largest possible whenever the first term in the sum interferes constructively with the other terms. This will happen whenever the first term is also real, and has the same sign as $b_1c_1(1-1/x)$. We can in fact achieve this by picking $\gamma = \vartheta + \pi$, since $0 < x < 1$. We obtain then that in our worst-case situation,

$$j_3 = \sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)} + b_1c_1(1/x - 1). \tag{5.14}$$

The Caves-Fuchs-Schack inequality is expressed then in our case as

$$j_3^2 + b_1^2 + c_1^2 + 2j_3b_1c_1 \leq 1, \tag{5.15}$$

where

$$x \in (0,1), r \in (0,\infty), b_1 \in \left[0, \sqrt{1/\left(1 + \frac{r}{x}\right)}\right), c_1 \in \left[0, \sqrt{1/\left(1 + \frac{1}{rx}\right)}\right), \tag{5.16}$$

and $j_3$ is given in (5.14). We will refer from now on to the left hand side of (5.15) as $f(x, r, b_1, c_1)$. If $b_1 = 0$ or $c_1 = 0$, a simple algebraic manipulation of the value of $j_3$ gives us that $f(x, r, b_1, c_1) \leq 1$. Expanding the value of $j_3$, we have that $f(x, r, b_1, c_1)$ is given by

$$\begin{aligned}
&b_1^2c_1^2(1 + 1/x^2 - 2/x) + \left(1 - b_1^2\left(1 + \frac{r}{x}\right)\right)\left(1 - c_1^2\left(1 + \frac{1}{rx}\right)\right) \\
&+ 2b_1c_1(1/x - 1)\sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)} \\
&+ b_1^2 + c_1^2 + 2b_1^2c_1^2(1/x - 1) + 2b_1c_1\sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)} \\
=&1 - b_1^2\frac{r}{x} - c_1^2\frac{1}{rx} + c_1^2b_1^2\left(\frac{2}{x^2} + \frac{1}{x}\left(r + \frac{1}{r}\right)\right) \\
&+ 2b_1c_1\frac{1}{x}\sqrt{1 - b_1^2\left(1 + \frac{r}{x}\right)}\sqrt{1 - c_1^2\left(1 + \frac{1}{rx}\right)}.
\end{aligned}$$

To prove that this is less or equal than 1, we will move all terms to one side of the inequality and write as a square the resulting expression (as an aside, this is a choice of approach for which we took inspiration from one of the standard proofs for the statement $x + \frac{1}{x} \geq 2$).

In more detail, multiplying by $x$ and dividing by $b_1^2 c_1^2$ our last expression, we have that $f(x, r, b_1, c_1)$ will be less or equal than 1 whenever

$$2\sqrt{\frac{1}{b_1^2} - \left(1 + \frac{r}{x}\right)}\sqrt{\frac{1}{c_1^2} - \left(1 + \frac{1}{rx}\right)} \leq r\frac{1}{c_1^2} + \frac{1}{r}\frac{1}{b_1^2} - \left(\frac{2}{x} + \left(r + \frac{1}{r}\right)\right) \tag{5.17}$$

Observe now that both sides of this inequality are positive. This is trivial for the left hand side, and follows for the right hand side from the previous obtained upper bounds on $b_1$ and $c_1$. If we square both sides of this inequality and simplify the resulting expression, we obtain

$$\left(r^2\left(\frac{1}{c_1^4} - \frac{2}{c_1^2} + 1\right) + \frac{1}{r^2}\left(\frac{1}{b_1^4} - \frac{2}{b_1^2} + 1\right) + 2\left(\frac{1}{c_1^2} + \frac{1}{b_1^2} - \frac{1}{b_1^2 c_1^2} - 1\right)\right) \geq 0. \tag{5.18}$$

This can be rewritten as

$$\left(r\left(\frac{1}{c_1^2} - 1\right) - \frac{1}{r}\left(\frac{1}{b_1^2} - 1\right)\right)^2 \geq 0, \tag{5.19}$$

which is true, so we have successfully proved that $a$, $b$ and $c$ satisfy the Caves-Fuchs-Schack inequality, and therefore can be excluded by a projective measurement.

Note that $x$ is not involved at all in (5.18), although one can verify computationally that the difference between the left hand side and the right hand side of (5.17) does depend on $x$.

## 5.3 Perspectives for generalization

### 5.3.1 Usage of Quadratically Constrained Quadratic Programs (QCQPs)

We will now discuss how to study the perfect exclusion of $n$ pure states by a projection through a collection of Quadratically Constrained Quadratic Programs (QCQPs). For a

situation with a $n$-dimensional complex variable $x$ and $m$ constraints, the standard form for such a program can be taken to be

$$\begin{aligned} &\text{minimize:} && x^* G x \\ &\text{subject to:} && x^* C_k x \geq l_k, \forall k \in \{1, \ldots, m\}, \end{aligned} \tag{5.20}$$

where the $l_k$ take real values, and $G$ and the $C_k$ are $n \times n$ Hermitian matrices.

This is a type of mathematical optimization formalism that has received considerable attention in recent years, with wide-ranging applications in science and engineering (see [4, 55, 65, 25] for just a few amongst many relevant examples). There has also been a considerable number of results about the theoretical structure of the corresponding problems and the design of algorithms that can solve them (see e.g. [62, 58, 83]). However, there have only been a handful of applications so far [66, 38, 11, 104] of the QCQP model to quantum information processing.

In our collection of QCQPs, there will be one program for every $n$-combination with repetition $\{s_1, \ldots, s_n\}$ out of the set $\{w_1, \ldots, w_n\}$ of states to be excluded. Each choice represents a possibility for how the states excluded after obtaining different outcomes of the projection relate to each other, and the reason why we need to consider those choices is that two different outcomes of the projection could plausibly lead to excluding the same state (which in the POVM case would be handled by grouping those two outcomes into the same one). In particular, each of the corresponding QCQPs for perfect state exclusion via projections formalizes the following two ideas:

- A projection in $n$ dimensions corresponds to a choice of $n$ unit vectors $\{v_1, \ldots, v_n\}$ that are pairwise orthogonal.
- We would like for every $v_i$ to be orthogonal to the corresponding $s_i$.

These ideas are then reflected in the following QCQP:

$$\begin{aligned} &\text{minimize:} && \sum_i v_i^* (s_i s_i^*) v_i \\ &\text{subject to:} && v_j^* v_k + v_k^* v_j = 0, \forall i, j \in \{1, \ldots, n\} \; s.t. \; j < k \\ & && i v_j^* v_k - i v_k^* v_j = 0, \forall j, k \in \{1, \ldots, n\} \; s.t. \; j < k \\ & && v_j^* v_j = 1, \forall j \in \{1, \ldots, n\}, \\ & && v_j \in \mathbb{C}^n, \forall j \in \{1, \ldots, n\}. \end{aligned} \tag{5.21}$$

Note that we have written $v_j^* v_k + v_k^* v_j = 0$ and $i v_j^* v_k - i v_k^* v_j = 0$ rather than $v_j^* v_k = 0$, in order to have the matrix representing each constraint be Hermitian, as required in (5.20) (one can then go as usual from an equality with 0 constraint to two constraints of inequality with respect to 0). We can also write $v_j^* v_j \geq 1$ rather than $v_j^* v_j = 1$, making usage of the fact that such a change does not alter whether the value of the program is 0 or not. Also, while for ease of presentation we have stated the problem with $n$ variables, they can be easily combined into one single variable taking values in $\mathbb{C}^{n^2}$ in order to obtain a program of the exact same form as (5.20).

The number of such programs in dimension $n$ that we need to consider is given by the number of $n$-combinations with repetition out of a set of length $n$, equal to $\binom{2n-1}{n}$. While asymptotically this will scale very quickly as a function of $n$, it will still be computationally tractable for values like $n = 5$ or $n = 6$, which goes beyond the theoretically understood range of up to $n = 3$. To compute the final answer, one will take the minimum value out of all the programs. If this value is equal to zero, then the states $\{w_1, \ldots, w_n\}$ can be perfectly excluded with a projection, while if it is a non-zero value then perfect state exclusion of the set $\{w_1, \ldots, w_n\}$ will not be possible.

As for its applications to future results, there are two main consequences of the formalism we just described, beyond the indirect consequence of our work possibly inspiring further usage of the QCQP framework within quantum information processing.

The first of these consequences correspond to our newfound ability to use results about QCQPs in order to obtain new structural results about the perfect exclusion of pure states through projections. One can straightforwardly check that basic weak duality results will not help, since the value of the Lagrangian dual programs will always be zero. However, as we discussed earlier there is an ongoing stream of non-trivial theoretical results about QCQPs, and it seems reasonable to conjecture that some of those results will eventually apply to the highly structured programs that we consider.

The second of these consequences corresponds to the increased potential for the usage of standard mathematical optimization packages. While the work on solver software supporting QCQP is not yet at a stage giving a simple path for an implementation of the programs described by (5.21), it seems reasonable to expect that such a stage will be reached in the near term. Then, such a piece of software could be compared with another one that implements the program in (5.1). From this, one would obtain a numerical study through standard solvers of the difference between POVMs and projections for perfect state exclusion of $n$ pure states in $n$ dimensions.

### 5.3.2 Direct generalizations of our proof

A naive approach for generalizing our result would start by considering conditions equivalent to the Caves-Fuchs-Schack inequality in the 4-dimensional case. However, this seems far from trivial, since the original derivation in [30] presents obstacles to such a generalization. In particular, it relies on the fact that when using the basis determined by an excluding projection, the sums corresponding to the inner products between two of the perfectly excluded states $\{a, b, c\}$ will have exactly one non-zero term. This makes it relatively easy to obtain formulas for the coefficients of $a$, $b$ and $c$ in that basis as a function of the inner products between the states. However, solving the corresponding equations in 4 dimensions seems like a significantly more complicated task, as each inner product between excluded states involves not 2 but 4 non-zero coefficients.

It could also be fruitful to take a geometrical perspective in order to better understand the situation at hand, following the approach in [15]. To see at an intuitive level what this might be like, one can start by observing that the space of density matrices is a section of the convex cone of positive semidefinite matrices. Also, the space of probability distributions with 3 outcomes can be seen as an equilateral triangle, with each vertex of the triangle corresponding to a different deterministic distribution. Then, as one can see in Chapter 10 of [15], for any fixed 3-outcome POVM the map which takes a density matrix to the probability distribution associated with applying the POVM to the density matrix will be an affine map from the convex cone section to the equilateral triangle.

In light of these facts, we can interpret any limits to state exclusion via projectors as saying that three points close to each other in the section of the convex cone cannot be sent to 3 different faces of the triangle by an affine map corresponding to a projection, as otherwise some points in the section would be sent outside the triangle, which is not allowed. Then, our result that projections are equivalent to POVMs can be seen as saying that in the case of pure states this does not change when we also allow the affine maps corresponding to non-projection POVMs. It might be interesting to fully formalize this thought, mathematically prove in this framework the known results about limits to state exclusion, and see if it is now easier to extend them to the case of 4 pure states, where the space of outcomes of a 4-outcome POVM can be seen as a regular tetrahedron.

Another way in which a geometric perspective might useful would be for obtaining a constructive algorithm that transforms an excluding POVM into an excluding projection for the case we analyze in this paper (3 pure states in 3 dimensions). It seems plausible that obtaining such an algorithm would then give insight about how to generalize our result. Note too that the main insight that leads to our result is the fact that one can take a POVM for perfect state exclusion to be an extremal one. This does not trivially

lead to an answer, since there are extremal POVMs that are not projections, such as those in the family in Equations (5.10)–(5.12). However, an analysis of what the ranks of an extremal POVM in 3 dimensions have to look like allows us to obtain a parametrization of the situation that can be algebraically solved. The usage of more sophisticated facts about the structure of extremal POVMs (such as those facts derived in [84, 48, 99]) could be similarly involved in a generalization of our results to higher dimensionality. In fact, these considerations seem to us a very likely ingredient of any such generalization.

It also a possibility is to look for partial characterizations for the perfect state exclusion of pure states (by projections or by a POVM) rather than for an exact characterization like the Caves-Fuchs-Schack. One could for example make progress along this path by proving the following conjecture from [49], and then determining whether it holds for the case of projective measurements:

**Conjecture 1** ([49])**.** *: If $n$ states $\{v_1, \ldots, v_n\}$ in $n$ dimensions are such that $|\langle v_i, v_j\rangle| < \frac{n-2}{n-1}$, it is possible to perform perfect state exclusion of these states with a POVM.*

Along the lines of using state exclusion characterizations alternative to the one given by (5.1), the work in [51] considers a generalization of the explicit perfect state exclusion criteria given in [30] for the 2-dimensional case. However, it finds this generalization to be a sufficient condition for the $n$-dimensional case but not a necessary one. This work also observes that if pure states are perfectly excluded via a POVM, they will be an eigenvector (with eigenvalue 0) of the corresponding POVM element, and they can be assumed to be an element of its spectral decomposition. Then, one can consider the feasibility of an optimization program where one tries to fill in the remaining coefficients and vectors in the spectral decomposition of the POVM elements. While one might expect at first glance that the standard SDP framework in (5.1) would offer a greater chance of applying mathematical optimization results, perhaps the fact that this formulation is closer in its shape to the QCQPs in (5.21) could help make non-trivial connections between the projection case and the POVM case.

### 5.3.3 Other considerations

It might also be of interest to find relations between the optimality of projections for tasks involving POVMs, and the optimality of unitaries (without the use of ancillas) for certain tasks involving channels, discussed for example in [20, 10], specially considering the numerical evidence suggestive of such kind of connection identified in [12, 10]. When doing

so, it might also be of interest to consider results (such as those in [39]) that characterize from a computational complexity point of view the power of computing with unitaries as opposed to general quantum channels.

Note too that for the exclusion of $k$ mixed states $\{\rho_1, \ldots, \rho_k\}$ in $n$ dimensions, the $k$-outcome POVM M that minimizes $\sum_i \langle M_i, \rho_i \rangle$ will be the one that maximizes $\sum_i \left\langle M_i, \frac{\mathcal{I}-\rho_i}{n-1} \right\rangle$ – that is, the one that performs best in the state discrimination of states $\{\frac{\mathcal{I}-\rho_1}{n-1}, \ldots, \frac{\mathcal{I}-\rho_k}{n-1}\}$. One can then apply existing results about optimal measurements for state discrimination [37, 41], while also considering the chance of generalizing results [36, 92] that look into the optimality of projections for state discrimination of pure states. Relatedly, as we discussed earlier the work in [30] gives a characterization for perfect exclusion of 2-dimensional pure states that makes it clear that for any number of $k > 2$ states in 2 dimensions, projections are not necessarily equivalent to POVMs. It is the case that they additionally use a reduction to that setting to point out that for three mixed states in three dimensions, projections are not equivalent to POVMs within the context of perfect state exclusion.

Note as well that if one considers the possibility of constraining the number of non-zero components of a perfectly excluding POVM, the possibility of doing so simply corresponds to being able to perfectly exclude a subset of the set of states under consideration. Another related variation one might want to study is requiring that there are no zero components of a perfectly excluding POVM, as considered in [51].

One could also look into determining whether the results here carry over to the gradual measure of PP-incompatibility defined in [27], which is the value of the SDP in (5.1) when the uniform distribution is assumed. This would correspond for example to asking whether projections are optimal for state exclusion of 3 pure states in 3 dimensions even when perfect exclusion cannot be achieved by POVMs. If that was successfully answered, it would be natural to relax assumptions even further, and consider arbitrary distributions. [89] offers a partial answer to these questions, by giving for an arbitrary number of pure states a sufficient condition for the existence of an optimal excluding POVM that is a projection (this is the condition that there is an optimal POVM such that none of the outcomes are perfectly excluded, and we also have that the pure states are linearly independent).

Note that the QCQP framework discussed in Section 5.3.1 extends without issues to the variants of the problem discussed in the previous paragraphs. In particular, if one wishes to study the exclusion of $k \neq n$ states, one can simply write $\{w_1, \ldots, w_k\}$ rather than $\{w_1, \ldots, w_n\}$ for the set of states to be excluded, giving rise to $\binom{k+n-1}{n}$ (rather than $\binom{2n-1}{n}$) programs of the form in (5.21). Similarly, if one wishes to study mixed states rather than pure states or introduce a probability distribution on the states, one can simply modify the objective function in (5.21) by replacing the values of $s_i s_i^*$ with a corresponding $\rho_i$ and

combining them with a multiplicative term $p_i$, respectively.

Furthermore, if one wishes to study non-perfect state exclusion, the programs given in Section 5.3.1 can be used towards that purpose without additional modifications. As for the variant that limits the number of non-zero components of a perfectly excluding POVM, it will correspond to limiting the number of distinct terms that can appear in the $n$-combinations with repetition of $\{w_1, \ldots, w_n\}$ that characterize the programs in (5.21). Similarly, the requirement that there are no zero components of the POVM corresponds to requiring that every state is excluded by a measurement outcome, and therefore to only considering the $n$-combination given by $\{s_1, \ldots, s_n\} = \{w_1, \ldots, w_n\}$.

# Chapter 6

# Quantum hedging

## 6.1 Setting

The results in this chapter concern the setting where a verifier Alice and a prover Bob conduct several instances of the same two-round protocol in parallel. This two-round protocol will go as follows:

1. Alice prepares a quantum state $\rho$, and sends part of this state to Bob, while keeping the other part herself.
2. Bob acts with a quantum channel $\Phi$ of his choice on the part of the state that he receives, and sends the answer to Alice.
3. Alice measures the quantum system under her control (i.e. Bob's answer and the share of the state $\rho$ that she kept) through a POVM $\{P_0, P_1\}$. If she obtains the outcome corresponding to $P_1$, we say that Bob "won", and if she obtains the outcome corresponding to $P_0$, we say that Bob "lost".

### 6.1.1 Background and motivation

In previous work published in [77], we provided the first know protocol of this kind where Bob can make sure that he obtains the winning outcome in one out of two parallel instances with probability 1 by correlating his behavior, while his optimal probability of winning one

instance is less than 1. We name this as perfect *quantum hedging*. The provided protocol corresponds to the simple case where

$$\rho = \frac{1}{2}(|00\rangle + |11\rangle)(\langle 00| + \langle 11|) \tag{6.1}$$

$$P_1 = (\cos(\pi/8)\,|00\rangle + \sin(\pi/8)\,|11\rangle)(\cos(\pi/8)\,\langle 00| + \sin(\pi/8)\,\langle 11|) \tag{6.2}$$

$$P_0 = \mathcal{I} - P_1 \tag{6.3}$$

and Alice sends to Bob one of the qubits in the standard maximally entangled state corresponding to $\rho$, while Bob also answers with a qubit. In a single instance, Bob's optimal chance of winning is $\cos(\pi/8)^2 < 1$, while in two parallel instances, he can win one out of the two with certainty by applying the phase flip

$$|00\rangle \mapsto -|00\rangle\,, |01\rangle \mapsto |01\rangle\,, |10\rangle \mapsto |10\rangle\,, |11\rangle \mapsto |11\rangle \tag{6.4}$$

to the two qubits that he receives, and then returning them back to Alice.

As pointed out in the original presentation of this result, one can consider a fictitious scenario to see why this type of hedging behavior can be of interest to Bob. In our scenario, Bob is offered the opportunity to take part in two potentially very lucrative (but involving some risks) games of chance, organized by Alice. These two games are completely identical to each other, and run independently. To earn the right to play in each of the games, Bob must contribute \$1 million of his own money, and he has an 85% chance of winning if he plays optimally. For each game he wins, Bob receives a price of \$3 million, with a total \$2 million gain over his initial investment. If Bob does not win, he loses his \$1 million investment.

Many people, if put in the place of Bob, would not hesitate to play both of the games, even taking out a \$2 million loan if necessary to do so. The expected gain from each of the games is \$1,550,000, and the only time that Bob loses money as an overall result is when Bob loses in both of the games. If we treat the games independently, the chance for a loss in both is 2.25%. However, Bob could be a highly risk-averse person. He would greatly enjoy being a millionaire, but cannot or does not want to risk a 2.25% chance of losing \$2 million. If the games run by Alice can be modelled classically, there is no way Bob can avoid this risk. However, if the two games have a model using quantum information with the same properties as the one in our protocol, Bob can be guaranteed to win in at least one of the games, and therefore obtain at least a total \$1 million gain. A choice of an appropriate quantum strategy allows Bob to hedge his bets perfectly.

The PhD work presented in this chapter has as its main goal to deepen our understanding of the quantum hedging phenomenon. The main result for this chapter is Theorem 3,

described in Section 6.2. This result had its original publication in [10], where it is part of a tight characterization of Bob's chances of winning at least 1 out of $n$ parallel instances for a generalization of the protocol from [77] that we just described. The purpose of the result is to better understand what kind of situation gives rise to the hedging phenomenon. In Section 6.3, we further discuss the combination of this hedging phenomenon with the usage of the logarithmic utility principle, common in the financial analysis of hedging decisions. In Claim 1 and its proof, we provide an example of how the application of this principle can motivate the search for better parameters in quantum hedging protocols.

In terms of length, the bulk of this chapter will be of a technical character, and concerned with the proofs of Theorem 3 in the case of Section 6.2, and Claim 1 in the case of Section 6.3. These proofs are non-trivial, and might be of inspiration to others who are in the process of construction non-trivial proofs in the setting of (quantum) prover-verifier interactions with a single prover. Readers who find it preferable at this time to skip the technical content are free to focus on the statement of Theorem 3 and Claim 1, together with the surrounding discussions that provide the appropriate context.

We now discuss why this quantum hedging phenomenon is of interest, and therefore worth being understood in a deeper way.

One key reason for the interest of quantum hedging is that this effect cannot be replicated in the classical counterpart of our setting, where it will be optimal for Bob to play independently between parallel instances. That is to say, assume that the messages exchanged between Alice and Bob are classical, and the optimal chance for Bob to win a single instance is equal to $p$. Then, Bob's optimal chance of winning at least $k$ out of $n$ parallel instances will be given by $\sum_{t=k}^{n} \binom{n}{t} p^t (1-p)^{n-t}$, and this will be the case even if the protocol has more than 2 rounds of communication (the situation does becomes more complex if there are several provers, see [40, 53, 94]).

The existence of quantum hedging establishes then a separation between the classical and quantum realms in a prover-verifier setting with a single prover, which can be contrasted with well-known separations [32, 72, 88] in the nonlocal games setting where the messages exchanged are classical and there are multiple provers. This creates then the potential of experiments based on single prover-verifier protocols that give rise to situations that cannot be mathematically explained with a non-quantum model under reasonable assumptions.

Beyond the opportunities for exploring this new kind of classical-quantum separation, there are at least two additional reasons to care about quantum hedging, corresponding respectively to our main result in Section 6.2 and the additional discussion in Section 6.3.

The first of these reasons is that the existence of the quantum hedging phenomenon

removes the possibility for a naive proof of the effectiveness of parallel repetition for error reduction in the complexity class QIP(2). This complexity class is studied for example in [95, 115, 56, 50], and corresponds to proof systems with the two-message interaction pattern here. If it was optimal to play independently in order to win at least $k$ out of $n$ parallel instances of a QIP(2) protocol, the standard proof in the classical case (see e.g. [5]) regarding the usage of parallel repetition for error reduction would apply. As found in later Masters work [75], there are however limits to the obstacles that quantum hedging presents to such a proof. In particular, there is a lack of ability to use quantum hedging to increase the expected number of parallel instances won when compared with playing independently. In light of this situation, we conducted the research presented in Section 6.2 with the purpose of gaining a better intuition of quantum hedging, and therefore of the exact obstacles that it might present to such a proof.

These obstacles were later surmounted in work by Hornby [54], who obtained bounds to quantum hedging that establish the effectiveness of parallel repetition for error reduction in quantum interactive proof systems with an arbitrary number of messaging rounds. In particular, these bounds establish that only $O(\log(1/\delta)\log(\log(1/\delta)))$ parallel instances of an interaction are needed to bring both the soundness and completeness error below a target value of $\delta$ via strong parallel repetition.

A second reason corresponds to a more general motivation for characterizing hedging behavior in any kind of protocol where it is possible. While the game scenario we earlier described is fictitious, it is well-understood in mathematical finance that there are practical situations, such as the presence of compound interest, where it is advantageous to engage in hedging behavior rather than in naive expected-value optimization (with this line of thought leading to the name of hedge funds for certain investment funds that use sophisticated portfolio construction techniques).

One technique to identify when to engage in hedging behavior is to make usage of the logarithmic utility principle [45]. It is then of interest to apply this principle in order to better understand the situations in which quantum hedging would be rational, as we do in our demonstration in Section 6.3. The practical application of this type of analysis is not far-fetched, given the recent push towards the large-scale implementation of quantum communication technologies [116], and parallel developments in electronic financial protocols [63, 74] that make usage of primitives that correspond to prover-verifier interactions.

### 6.1.2 Semidefinite programming formulation

Consider $n$ parallel instances of a two-round protocol of the type considered here, with Alice initially preparing the state $\rho$, and measuring with the POVM $\{P_0, P_1\}$ to determine the final outcome. Denote the Hilbert space for Alice's message to Bob in the $i^{th}$ instance of the game as $\mathcal{X}_i$. Similarly, denote the Hilbert space for Bob's message to Alice in the $i^{th}$ instance of the game as $\mathcal{Y}_i$.

It follows from the general formalism presented in [46, 47] that the optimal chance for Bob to win at least 1 out of $n$ parallel repetitions will be given by the following semidefinite program:

$$\underline{m_{n,\alpha,\theta}\text{: Primal problem}}$$

$$\begin{aligned} \text{minimize:} \quad & \left\langle Q_0^{\otimes n}, X \right\rangle \\ \text{subject to:} \quad & \mathrm{Tr}_{\mathcal{Y}_1 \otimes \cdots \otimes \mathcal{Y}_n}(X) = \mathcal{I}_{\mathcal{X}_1 \otimes \cdots \otimes \mathcal{X}_n}, \\ & X \in \mathrm{Pos}(\mathcal{Y}_1 \otimes \mathcal{X}_1 \otimes \cdots \otimes \mathcal{Y}_n \otimes \mathcal{X}_n) \end{aligned} \tag{6.5}$$

$$\underline{m_{n,\alpha,\theta}\text{: Dual problem}}$$

$$\begin{aligned} \text{maximize:} \quad & \mathrm{Tr}(Y) \\ \text{subject to:} \quad & \pi \left( \mathcal{I}_{\mathcal{Y}_1 \otimes \cdots \otimes \mathcal{Y}_n} \otimes Y \right) \pi^* \leq Q_0^{\otimes n}, \\ & Y \in \mathrm{Herm}(\mathcal{X}_1 \otimes \cdots \otimes \mathcal{X}_n) \end{aligned} \tag{6.6}$$

where:

- $\pi$ is a unitary operator that permutes the order of Hilbert spaces in a tensor product, with the action

  $$\pi(y_1 \otimes \cdots \otimes y_n \otimes x_1 \otimes \cdots \otimes x_n) = y_1 \otimes x_1 \otimes \cdots \otimes y_n \otimes x_n.$$

  for any $y_1 \in \mathcal{Y}_1, \cdots, y_n \in \mathcal{Y}_n$ and $x_1 \in \mathcal{X}_1, \cdots, x_n \in \mathcal{X}_n$

- $Q_0 = \left( \mathcal{I}_{\mathcal{Y}_i} \otimes \Psi_\rho \right)(P_0)$, with $\Psi_\rho$ the channel such that $J(\Psi_\rho) = \overline{\rho}$.

In [46, 47], one can see more general semidefinite programs corresponding to Bob winning at least $k$ out of $n$ parallel instances, but this 1-out-of-$n$ program will suffice for the purpose of our analysis in this chapter.

## 6.2 Main result

### 6.2.1 Formal statement of main result

As discussed in Section 6.1.1, we aim here to derive results that can be used to build intuition about the situations where quantum hedging can occur, as well as the magnitude of the corresponding advantages. In particular, we study a generalization of the protocol in [77] and derive the following theorem, whose proof appears in Section 6.2.2:

**Theorem 3.** *Consider $n$ parallel instances of a protocol where:*

1. *Alice prepares a quantum state $\rho = |\phi\rangle\langle\phi|$ where $|\phi\rangle = \alpha|00\rangle + \sqrt{1-\alpha^2}|11\rangle$ with $\alpha \in [0,1]$, and sends one qubit to Bob, while keeping the other qubit herself.*

2. *Bob acts with a qubit channel $\Phi$ on the qubit that he receives, and sends the resulting qubit back to Alice.*

3. *Alice measures the two qubits under her control through a POVM $\{P_0 = |\psi\rangle\langle\psi|, P_1 = \mathcal{I} - P_0\}$ in order to determine whether Bob win or loses, with $|\psi\rangle = \cos(\theta)|00\rangle + \sin(\theta)|11\rangle$.*

*Note that in these $n$ parallel instances, Bob is not constrained to channels of the form $\Phi^{\otimes n}$ that treat the qubit in each instance identically and independently. Instead, he can apply any channel that maps $n$ qubits to $n$ qubits.*

*Let $\theta_{n,\alpha} = \tan^{-1}\left(\sqrt{\frac{1}{\alpha^2}-1}\left(2^{1/n}-1\right)\right)$, and $\gamma_{n,\alpha} = \tan^{-1}\left(\sqrt{\frac{1}{\alpha^2}-1}\left(\frac{1}{2^{1/n}-1}\right)\right)$.*

*Then, if $\theta \in [0, \theta_{n,\alpha})$, the optimal chance for Bob to win at least 1 out of $n$ parallel instances will be given by $1 - \frac{1}{2^n}\left(\frac{1}{2}\right)^{n/2}(2\cos(\theta)^n - (\cos(\theta)+\sin(\theta))^n)$.*

*If $\theta \in (\gamma_{n,\alpha}, \pi/2]$, the optimal chance for Bob to win at least 1 out of $n$ parallel instances will be similarly given by $1 - \frac{1}{2^n}\left(\frac{1}{2}\right)^{n/2}(2\sin(\theta)^n - (\cos(\theta)+\sin(\theta))^n)$.*

*Furthermore, one can obtain an optimal channel for Bob by considering the unitary channels $\Phi_\Lambda(X) = \Lambda_n X \Lambda_n^*$ and $\Phi_\Xi(X) = \Xi_n X \Xi_n^*$ respectively, where:*

$$\Lambda_n = \sum_{r\in\{0,1\}^n} (-1)^{\wedge r + \oplus r} |r\rangle\langle r|$$

$$\Xi_n = \sum_{r\in\{0,1\}^n} (-1)^{\vee r + \oplus r} |r\rangle\langle r|.$$

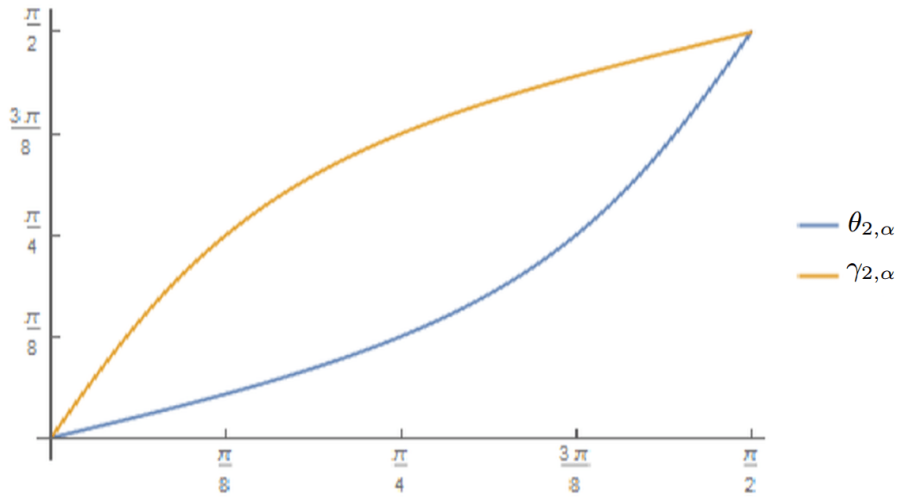


Figure 6.1: Illustration of the quantities involved in Theorem 3, for the case where $n = 2$.

One can see the general shape of $\theta_{n,\alpha}$ and $\gamma_{n,\alpha}$ as a function of $\alpha$ in Figure 6.1, which depicts the values of $\theta_{2,\alpha}$ and $\gamma_{2,\alpha}$. The curves would be of a similar shape for larger values of $n$, but with an increasing size of the region between the two lines.

This theorem characterizes the optimal chance for Bob to win at least 1 out of $n$ parallel instances in the protocol under study, together with other results also appearing in [10] that correspond to the case where $\theta \in [\theta_{n,\alpha}, \gamma_{n,\alpha}]$ and give Bob a perfect chance of doing so in that case.

Note that even in the $\theta \in [0, \theta_{n,\alpha}) \cup (\gamma_{n,\alpha}, \pi/2]$ range that we analyze here with the conclusion that perfect hedging is not possible, Bob benefits from not playing independently. For example, in the $\theta \in [0, \theta_{n,\alpha})$ case, one obtains from Theorem 3 that Bob's optimal advantage from correlating his answers is equal to

$$\left(1 - \frac{1}{2^n}\left(\frac{1}{2}\right)^{n/2}\left(2\cos(\theta)^n - (\cos(\theta) + \sin(\theta))^n\right)\right) - \left(1 - \frac{1}{2^n}\left(\frac{1}{2}\right)^{n/2}(\cos(\theta) - \sin(\theta))^n\right)$$

$$= \frac{1}{2^{n-1}}\left(\frac{1}{2}\right)^{n/2}\left(\sum_{\substack{2 \leq a \leq n \\ a \text{ even}}} \binom{n}{a} \sin(\theta)^a \cos(\theta)^{n-a}\right).$$

### 6.2.2 Proof of Theorem 3

*Proof.* We will consider the case where $\theta < \theta_{n,\alpha}$. The other case proceeds similarly. To simplify our argument, we will occasionally incur in notation abuse in this proof, and omit the permutation operators in the definition of the dual SDP (6.6) that remind us that matrices at the sides of a $\leq$ inequality must have their entries reordered to make the spaces on which they are defined be in the same order at both sides of the inequality. We will also assume for convenience that $\alpha \in (0,1)$ – the cases where $\alpha = 0$ or $\alpha = 1$ can be dealt with by continuity with respect to the $(0,1)$ range.

To prove that perfect hedging is not possible when $\theta < \theta_{n,\alpha}$, we prove the feasibility in the dual SDP (6.6) of an operator $Y$ with positive objective value. This operator is obtained from applying complementary slackness conditions to the primal solution corresponding to the strategy where Bob applies the unitary operation $\Lambda_n$ to his qubits. Therefore, it has value for the dual equal to the value in the primal SDP (6.5) for the solution corresponding to $\Lambda_n$. By weak duality, its feasibility proves then the optimality of $\Lambda_n$ when $\theta < \theta_{n,\alpha}$.

In order to make parameter dependance explicit, we will write $Q_{0,\alpha,\theta}$ for the value of $Q_0$ corresponding to a specific choice of $\alpha$ and $\theta$, as discussed in Section 6.1.2. A similar meaning is associated to the symbols $\rho_\alpha$, $P_{0,\theta}$, and $|\phi_\alpha\rangle$.

To prove the feasibility of $Y$, we will express $Q_{0,\alpha,\theta}^{\otimes n} - \pi\left(\mathcal{I}_{\mathcal{Y}_1\otimes\cdots\otimes\mathcal{Y}_n} \otimes Y\right)\pi^*$ as a direct sum of smaller matrices. This reduces the question about feasibility of $Y$ to a question about the positive-semidefiniteness of these smaller matrices. Each of these smaller matrices will have all proper leading principal minors be positive semi-definite, so by Sylvester's criterion it will suffice to check that their determinant is non-negative. We will then obtain a closed formula for these determinants, and prove that they are indeed non-negative.

We will first consider the case with $\alpha = 1/\sqrt{2}$, and then give an overview of the small changes involved in adapting the proof to other values of $\alpha$.

#### Study of $Q_{0,1/\sqrt{2},\theta}^{\otimes n}$

$Q_{0,\alpha,\theta} \in \mathrm{Pos}(\mathcal{X}\otimes\mathcal{Y})$ is given by $|\psi_0^1\rangle\langle\psi_0^1| + |\psi_0^2\rangle\langle\psi_0^2| + |\psi_0^3\rangle\langle\psi_0^3|$, where the $|\psi_0^i\rangle$ are defined as

$$\begin{aligned} \left|\psi_0^1\right\rangle &= \alpha\sin(\theta)\,|00\rangle - \sqrt{1-\alpha^2}\cos(\theta)\,|11\rangle\,, \\ \left|\psi_0^2\right\rangle &= \alpha\,|01\rangle\,, \\ \left|\psi_0^3\right\rangle &= \sqrt{1-\alpha^2}\,|10\rangle\,. \end{aligned} \tag{6.7}$$

This follows from considering the value of $P_{0,\theta}$, and observing that the operator $\Psi_{\rho_\alpha}$ satisfying $J(\Psi_{\rho_\alpha}) = \overline{|\phi_\alpha\rangle\langle\phi_\alpha|}$ (with $|\phi_\alpha\rangle = \alpha|00\rangle + \sqrt{1-\alpha^2}|11\rangle$ the initial state shared between Alice and Bob) maps a state $\sigma \in \mathrm{D}(\mathcal{Z})$ to $(\alpha|0\rangle\langle 0| + \sqrt{1-\alpha^2}|1\rangle\langle 1|)\sigma(\alpha|0\rangle\langle 0| + \sqrt{1-\alpha^2}|1\rangle\langle 1|)$. We can then write $Q^{\otimes n}_{0,1/\sqrt{2},\theta}$ as

$$\begin{aligned} Q^{\otimes n}_{0,1/\sqrt{2},\theta} = \quad & \left(\frac{1}{2}\right)^n \Big((\sin(\theta)|00\rangle - \cos(\theta)|11\rangle)(\sin(\theta)\langle 00| - \cos(\theta)\langle 11|) \\ & + |01\rangle\langle 01| + |10\rangle\langle 10|\Big)^{\otimes n} \qquad (6.8) \\ = \quad & \left(\frac{1}{2}\right)^n \sum_{a,b,c,d\in\{0,1\}^n} |a\rangle|b\rangle\langle c|\langle d| \prod_{i=0}^{n-1} \Big(\delta_{c_i,1-d_i}\delta_{a_i,c_i}\delta_{b_i,d_i} \\ & + \delta_{a_i,b_i}\delta_{c_i,d_i}\Big(\delta_{a_i,1-c_i}(-\sin(\theta)\cos(\theta)) + \delta_{a_i,c_i}\delta_{a_i,1}\cos(\theta)^2 \\ & + \delta_{a_i,c_i}\delta_{a_i,0}\sin(\theta)^2\Big)\Big) \\ = \quad & \left(\frac{1}{2}\right)^n \sum_{a,c\in\{0,1\}^n} |a\rangle\langle c| \otimes \sum_{b,d\in\{0,1\}^n} |b\rangle\langle d| \prod_{i=0}^{n-1} \Big(\delta_{a_i,1-b_i}\delta_{c_i,1-d_i}\delta_{a_i,c_i} + \\ & \delta_{a_i,b_i}\delta_{c_i,d_i}\Big(\delta_{a_i,1-c_i}(-\sin(\theta)\cos(\theta)) + \delta_{a_i,c_i}\delta_{a_i,1}\cos(\theta)^2 \\ & + \delta_{a_i,c_i}\delta_{a_i,0}\sin(\theta)^2\Big)\Big). \qquad (6.9) \end{aligned}$$

The key insight to go ahead with the proof is to notice that this matrix can be written as a direct sum of $3^n$ smaller matrices. Indeed, observe that (6.8) can be equivalently written as

$$\frac{1}{2^n} \sum_{w\in\{0,1,2\}^n} \bigotimes_{i=0}^{n-1} |\psi_{w_i}\rangle\langle\psi_{w_i}|, \text{ where } |\psi_{w_i}\rangle = \begin{cases} \sin(\theta)|00\rangle - \cos(\theta)|11\rangle, & \text{if } w_i = 0 \\ |01\rangle, & \text{if } w_i = 1 \\ |10\rangle, & \text{if } w_i = 2 \end{cases}. \qquad (6.10)$$

Then, the coefficient for each $|a\rangle\langle c| \otimes |b\rangle\langle d|$ term in the summation in (6.9) will receive

contribution from at most one of the elements in (6.10). This element will be the one where

$$w_i = \begin{cases} 0 \text{ if } a_i = b_i \\ 1 \text{ if } (a_i, b_i) = (0,1) \\ 2 \text{ if } (a_i, b_i) = (1,0). \end{cases} \tag{6.11}$$

Since this only depends on $|ab\rangle$, all elements on the same row of $Q^{\otimes n}_{0,1/\sqrt{2},\theta}$ come from the same term in (6.10). As each row of $Q^{\otimes n}_{0,1/\sqrt{2},\theta}$ has at least one non-zero term, (6.10) implies then a decomposition of $Q^{\otimes n}_{0,1/\sqrt{2},\theta}$ into a direct sum of smaller matrices, each of them with rank 1.

We can then identify each of these matrices by the corresponding choice of $w$ in (6.10). We will do so by writing them as $Q^{\otimes n}_{0,1/\sqrt{2},\theta}(w)$. We denote the number of 0s, 1s and 2s in $w$ by $n_0(w)$, $n_1(w)$ and $n_2(w)$, respectively. Also, note that there will be $3^n$ matrices in our decomposition, with the dimension of $Q^{\otimes n}_{0,1/\sqrt{2},\theta}(w)$ being given by $2^{n_0(w)}$. Note too that the number of matrices of size $2^k$ is given by $\binom{n}{k}2^{n-k}$. This corresponds to choosing on which $k$ positions $w_i = 0$, and what is the value of $w_i$ for the other ones.

It will be convenient later to have a formula for the restriction to the diagonal of $Q^{\otimes n}_{0,1/\sqrt{2},\theta}(w)$. Using the description in (6.10), we have that it is given by

$$\left(\frac{1}{2}\right)^n \sum_{w' \in M_w \subseteq \{0,1\}^n} g(w,w') \, |w'\rangle \, |f(w,w')\rangle \, \langle w'| \, \langle f(w,w')| \tag{6.12}$$

where $M_w$ is given by the cartesian product $\bigtimes_{i=0}^{n-1} M_{w_i}$, with $\begin{cases} M_0 = \{0,1\} \\ M_1 = \{0\} \\ M_2 = \{1\} \end{cases}$,

$g(w,w') = \prod_{i=0}^{n-1} g(w_i, w'_i)$ with $\begin{cases} g(0,0) = \sin^2(\theta) \\ g(0,1) = \cos^2(\theta) \\ g(1,0) = 1 \\ g(2,1) = 1 \end{cases}$, $f(w,w')_i = \begin{cases} w'_i \text{ if } w_i = 0 \\ 1 - w'_i \text{ if } w_i = 1 \end{cases}$.

Note that due to the definition of $M_w$, it is not necessary to define $g(w_i, w'_i)$ for the values of $(w_i, w'_i)$ not included in our definition of $g$ here.

**Presentation of our candidate for $Y$ in the $\alpha = 1/\sqrt{2}$ case**

We define now our candidate solution $Y$ for the dual problem, given by

$$Y = -\epsilon \left( \left( \frac{1}{\sqrt{2}} \sin(\theta) \left|0\right\rangle \left\langle 0\right| + \frac{1}{\sqrt{2}} \cos(\theta) \left|1\right\rangle \left\langle 1\right| \right)^{\otimes n} - 2 \left( \frac{1}{\sqrt{2}} \cos(\theta) \left|1\right\rangle \left\langle 1\right| \right)^{\otimes n} \right), \quad (6.13)$$

where $\epsilon$ is a value $> 0$ given by $\left(\frac{1}{2}\right)^{n/2} (2\cos(\theta)^n - (\cos(\theta) + \sin(\theta))^n)$. Note that the definition of $\theta_{n,1/\sqrt{2}}$ implies that this value is positive indeed for $\theta < \theta_{n,1/\sqrt{2}}$. We can then write $Y$ as

$$\sum_{a \in \{0,1\}^n} \lambda_a \left|a\right\rangle \left\langle a\right|, \text{where } \lambda_a = \begin{cases} -\epsilon \left(\frac{1}{2}\right)^{n/2} \sin(\theta)^{n-|a|} \cos(\theta)^{|a|} & \text{for } a \neq 1^n \\ \epsilon \left(\frac{1}{2}\right)^{n/2} \cos(\theta)^n & \text{for } a = 1^n \end{cases} \quad (6.14)$$

Note that its trace (i.e., its value for the dual program) is given by

$$-\left(\frac{1}{2}\right)^{n/2} \epsilon \Big( (\sin(\theta) + \cos(\theta))^n - 2\cos(\theta)^n \Big), \quad (6.15)$$

which will again be positive for $\theta < \theta_{n,1/\sqrt{2}}$ by definition of $\theta_{n,1/\sqrt{2}}$.

This $Y$ has been obtained from the strategy $\Lambda_n$ defined in Section 6.1, and its feasibility proves the optimality of $\Lambda_n$ for $\theta < \theta_{n,1/\sqrt{2}}$. This is an example of complementary slackness behavior, and follows from the observation [111] that given a feasible solution $X$ to the primal SDP (6.5), $\mathrm{Tr}_{\mathcal{Y}_1 \otimes \cdots \otimes \mathcal{Y}_n}(Q_{0,\alpha,\theta}^{\otimes n} X)$ is an operator with the same objective value for the dual SDP (6.6), and satisfies the feasibility constraints of the dual if and only if $X$ represents an optimal solution to the primal. Therefore, after we experimentally observed that $\Lambda_n$ seemed to be optimal for $\theta < \theta_{n,\alpha}$, we computed the corresponding value of $\mathrm{Tr}_{\mathcal{Y}_1 \otimes \cdots \otimes \mathcal{Y}_n}(Q_{0,1/\sqrt{2},\theta}^{\otimes n} X)$ to obtain our proposed $Y$. $X$ is given in this computation by the primal solution that represents the channel for the unitary in $\Lambda_n$,

$$X = \sum_{i,j \in \{0,1\}^n} \left|ii\right\rangle \left\langle jj\right| (-1)^{\wedge i + \bigoplus i + \wedge j + \bigoplus j}. \quad (6.16)$$

**Feasibility of $Y$ in the $\alpha = 1/\sqrt{2}$ case**

We want to prove that $Y$ is feasible – that is to say, $Q_{0,1/\sqrt{2},\theta}^{\otimes n} - Y \otimes \mathcal{I} \geq 0$. Since $Y$ is diagonal, the direct sum decomposition of $Q_{0,1/\sqrt{2},\theta}^{\otimes n}$ corresponds to a direct sum decomposition of $Y$. Since positive semidefiniteness is preserved by the direct sum operator, it is then enough to prove that each of the $S_w = Q_{0,1/\sqrt{2},\theta}^{\otimes n}(w) - (Y \otimes \mathcal{I})(w)$ matrices are positive semidefinite, where $(Y \otimes \mathcal{I})(w)$ denotes $Y \otimes \mathcal{I}$ restricted to the rows/columns of $Q_{0,1/\sqrt{2},\theta}^{\otimes n}$ assigned to $Q_{0,1/\sqrt{2},\theta}^{\otimes n}(w)$.

Consider first the largest of these matrices. This will be $S_{0^n}$, with size $2^n$. Using (6.10), we have that it is given by

$$S_{0^n} = \left(\frac{1}{2}\right)^n \sum_{a,c \in \{0,1\}^n} |aa\rangle \langle cc| \Big( \prod_{i=0}^{n-1} \Big( \delta_{a_i,1-c_i} \cdot -\sin(\theta)\cos(\theta) + \delta_{a_i,c_i}\delta_{a_i,1}\cos(\theta)^2 + \delta_{a_i,c_i}\delta_{a_i,0}\sin(\theta)^2 \Big) - 2^n \lambda_a \Big).$$

For example, for $n = 2$, $S_{00}$ is given by

$$\frac{1}{4}\begin{pmatrix} \sin(\theta)^4 - 4\lambda_{00} & -\sin(\theta)^3\cos(\theta) & -\sin(\theta)^3\cos(\theta) & \sin(\theta)^2\cos(\theta)^2 \\ -\sin(\theta)^3\cos(\theta) & \sin(\theta)^2\cos(\theta)^2 - 4\lambda_{01} & \sin(\theta)^2\cos(\theta)^2 & -\sin(\theta)\cos(\theta)^3 \\ -\sin(\theta)^3\cos(\theta) & \sin(\theta)^2\cos(\theta)^2 & \sin(\theta)^2\cos(\theta)^2 - 4\lambda_{10} & -\sin(\theta)\cos(\theta)^3 \\ \sin(\theta)^2\cos(\theta)^2 & -\sin(\theta)\cos(\theta)^3 & -\sin(\theta)\cos(\theta)^3 & \cos(\theta)^4 - 4\lambda_{11} \end{pmatrix}$$

Consider now that since $Q_{0,1/\sqrt{2},\theta}^{\otimes n} \geq 0$, and for $a \neq 1^n$, $\lambda_a < 0$, the first $2^n - 1$ principal minors of $S_{0^n}$ are $\geq 0$. By Sylvester's criterion, to prove that $S_{0^n} \geq 0$, it suffices then to prove that $\det(S_{0^n}) \geq 0$. Note that $\det(S_{0^n})$ is a polynomial in $\epsilon$. This polynomial has all the coefficients below the one for $\epsilon^{2^n-1}$ equal to 0. This is because $Q_{0,1/\sqrt{2},\theta}^{\otimes n}(0^n)$ has rank 1 - therefore, each minor of it with at least two rows will have determinant equal to zero. Using this, and going through the determinant formula, we see that $\det(S_{0^n})$ is given by

$$\left( \epsilon^{2^n-1}(-1)^{2^n-1} \sum_{a \in \{0,1\}^n} \left(\frac{1}{2}\right)^n \cos(\theta)^{2|a|} \sin(\theta)^{2(n-|a|)} \prod_{\substack{b \in \{0,1\}^n \\ b \neq a}} \frac{\lambda_b}{\epsilon} \right) + \left( \epsilon^{2^n}(-1)^{2^n} \prod_{a \in \{0,1\}^n} \frac{\lambda_a}{\epsilon} \right) \tag{6.17}$$

$$=\epsilon^{2^n-1}\left(\epsilon-\sum_{a\in\{0,1\}^n}\frac{\left(\frac{1}{2}\right)^n\cos(\theta)^{2|a|}\sin(\theta)^{2(n-|a|)}}{\lambda_a/\epsilon}\right)\prod_{a\in\{0,1\}^n}\frac{\lambda_a}{\epsilon} \tag{6.18}$$

$$=\epsilon^{2^n-1}\left(\epsilon+\sum_{a\in\{0,1\}^n}\left(\frac{1}{2}\right)^{n/2}\cos(\theta)^{|a|}\sin(\theta)^{n-|a|}-2\left(\frac{1}{2}\right)^{n/2}\cos(\theta)^n\right)\prod_{a\in\{0,1\}^n}\frac{\lambda_a}{\epsilon} \tag{6.19}$$

Since all of the $\lambda_a/\epsilon$ except the one for $1^n$ are negative, we have that the $\epsilon^{2^n-1}\prod_{a\in\{0,1\}^n}\frac{\lambda_a}{\epsilon}$ term is negative whenever $\epsilon>0$. Therefore,

$$\det(S_{0^n,0^n})\geq 0 \iff \tag{6.20}$$

$$\epsilon+\sum_{a\in\{0,1\}^n}\left(\frac{1}{2}\right)^{n/2}\cos(\theta)^{|a|}\sin(\theta)^{n-|a|}-2\left(\frac{1}{2}\right)^{n/2}\cos(\theta)^n\leq 0 \iff \tag{6.21}$$

$$\epsilon\leq\left(\frac{1}{2}\right)^{n/2}\left(2(\cos(\theta))^n-(\cos(\theta)+\sin(\theta))^n\right), \tag{6.22}$$

which is true by definition of $\epsilon$. We have then that our proposed feasible solution $Y$ produces a positive-semidefinite $S_{0^n}$. To verify the feasibility of $Y$, it remains to prove the positive-semidefiniteness of the rest of the $S_w$.

To do so, consider an arbitrary $S_w$, $w\in\{0,1,2\}^n-\{0^n\}$, with a corresponding $M_w$, as defined in (6.12). Note that $M_w$ is the set of indices i such that $\lambda_i$ appears in the diagonal of $S_w$, and that each $\lambda_i$ appears in the diagonal of $S_w$ at most once, as we can see from the expression in (6.12). If $1^n\notin M_w$, then $S_w$ is trivially positive-semidefinite, since it is obtained by adding a positive-semidefinite diagonal matrix $Y(w)$ to a positive-semidefinite matrix $Q^{\otimes n}_{0,1/\sqrt{2},\theta}(w)$. Otherwise, our appeal to Sylvester's criterion from the $0^n$ case applies again, and it is enough to prove that $\det(S_w)\geq 0$. Also, since $Q^{\otimes n}_{0,1/\sqrt{2},\theta}(w)$ has rank 1, our argument that $\det(S_w)$ is a polynomial of minimum degree $|M_w|-1$ applies again.

Then, using (6.12), and similar to the derivation for (6.17), we have that $\det(S_w)$ is given by

$$\epsilon^{|M_w|-1}\left(\prod_{c\in M_w}\frac{\lambda_c}{\epsilon}\right)\left(\epsilon-\left(\frac{1}{2}\right)^n\sum_{d\in M_w}\frac{g(w,d)}{\lambda_d/\epsilon}\right). \tag{6.23}$$

Using the definitions of $M_w$ and $g(w,d)$ in (6.12), and realizing that $1^n \in M_w$ implies that $n_1(w)=0$, we have that

$$\sum_{d\in M_w}\frac{g(w,d)}{|\lambda_d/\epsilon|}=\left(\frac{1}{2}\right)^{n/2}(\sin(\theta)+\cos(\theta))^{n_0(w)}\left(\frac{1}{\cos(\theta)}\right)^{n_2(w)}. \tag{6.24}$$

Now, we have that

$$\frac{1}{\cos(\theta)}\leq\sin(\theta)+\cos(\theta)\iff\frac{1}{\cos(\theta)^2}\leq\tan(\theta)+1 \tag{6.25}$$

$$\iff\tan(\theta)^2\leq\tan(\theta)\iff\theta\leq\pi/4. \tag{6.26}$$

Since we are looking at the range $\theta<\theta_{n,1/\sqrt{2}}\leq\pi/4$, and $n_0(w)+n_2(w)=n$, we have that

$$(\sin(\theta)+\cos(\theta))^{n_0(w)}\left(\frac{1}{\cos(\theta)}\right)^{n_2(w)}\leq(\sin(\theta)+\cos(\theta))^n. \tag{6.27}$$

Therefore, since $n_2(w)\leq n$,

$$\left(\frac{1}{2}\right)^n\sum_{d\in M_w}\frac{g(w,d)}{\lambda_d/\epsilon}\geq\left(\frac{1}{2}\right)^{n/2}\left(2(\cos(\theta))^n-(\cos(\theta)+\sin(\theta))^n\right). \tag{6.28}$$

We see then that any $\epsilon$ that makes $\det(S_{0^n})$ non-negative will make the determinant of the other $S_w$ non-negative as well.

**Generalization to $\alpha \neq 1/\sqrt{2}$**

For $\alpha \neq 1/\sqrt{2}$, the changes necessary to make the proof work are limited to arithmetic adjustments. $Q_{0,\alpha,\theta}^{\otimes n}$ will now be given by

$$\sum_{a,c\in\{0,1\}^n} |a\rangle\langle c| \otimes \sum_{b,d\in\{0,1\}^n} |b\rangle\langle d| \prod_{i=0}^{n-1} \Big(\delta_{a_i,1-b_i}\delta_{c_i,1-d_i}\delta_{a_i,c_i}\Big(\delta_{a_i,1}(1-\alpha^2)+\delta_{a_i,0}\alpha^2\Big)$$
$$+\delta_{a_i,b_i}\delta_{c_i,d_i}\Big(\delta_{a_i,1-c_i}\cdot -\alpha\sin(\theta)\sqrt{1-\alpha^2}\cos(\theta)+\delta_{a_i,c_i}\delta_{a_i,1}(1-\alpha^2)\cos(\theta)^2$$
$$+\delta_{a_i,c_i}\delta_{a_i,0}\alpha^2\sin(\theta)^2\Big)\Big). \tag{6.29}$$

Note that its direct sum decomposition is not affected, since the choice of which terms of $Q_{0,\alpha,\theta}^{\otimes n}$ appear on each term does not depend on $\alpha$.

Similarly, $Y$ is given now by

$$\sum_{a\in\{0,1\}^n} \lambda_a|a\rangle\langle a|, \text{ where } \lambda_a = \begin{cases} -\epsilon(\alpha\sin(\theta))^{n-|a|}\left(\sqrt{1-\alpha^2}\cos(\theta)\right)^{|a|} & \text{for } a \neq 1^n \\ \epsilon\left(\sqrt{1-\alpha^2}\right)^n\cos(\theta)^n & \text{for } a = 1^n \end{cases}$$
$$\text{and } \epsilon = 2\left(\sqrt{1-\alpha^2}\cos(\theta)\right)^n - (\sqrt{1-\alpha^2}\cos(\theta)+\alpha\sin(\theta))^n. \tag{6.30}$$

In order to determine the feasibility of $Y$, we have now that $\det(S_w)$ is given by

$$\epsilon^{|M_w|-1}\left(\prod_{c\in M_w}\frac{\lambda_c}{\epsilon}\right)\left(\epsilon - \sum_{d\in M_w}\frac{g(w,d)\alpha^{2(n-|d|)}(1-\alpha^2)^{|d|}}{\lambda_d/\epsilon}\right), \tag{6.31}$$

again non-negative whenever

$$\epsilon \leq \sum_{d\in M_w}\frac{g(w,d)\alpha^{2(n-|d|)}(1-\alpha^2)^{|d|}}{\lambda_d/\epsilon}$$
$$= 2\left(\sqrt{1-\alpha^2}\right)^n\cos(\theta)^{2n_0(w)-n} - \sum_{d\in M_w}\frac{g(w,d)\alpha^{2(n-|d|)}(1-\alpha^2)^{|d|}}{|\lambda_d|/\epsilon}. \tag{6.32}$$

Using the definitions in (6.12), we have now that

$$\sum_{d\in M_w} \frac{g(w,d)\alpha^{2(n-|d|)}(1-\alpha^2)^{|d|}}{|\lambda_d|/\epsilon} = (\alpha\sin(\theta)+\sqrt{1-\alpha^2}\cos(\theta))^{n_0(w)}\left(\frac{\sqrt{1-\alpha^2}}{\cos(\theta)}\right)^{n_2(w)} \tag{6.33}$$

To prove that (6.32) holds we will need an argument slightly more involved than the corresponding one for the $\alpha = \frac{1}{\sqrt{2}}$ case. First, we consider that for $n_0(w) = n$, the right hand side of (6.32) is equal to $\epsilon$, by the expression in (6.33) and the definition of $\epsilon$ in (6.30). Then, we prove that the right hand side of (6.32) increases as we decrement $n_0(w)$, and increase $n_2(w) = n - n_0(w)$ in parallel. This is because the positive term in the right hand side increases with each decrease of $n_0(w)$, and it does so by a larger factor than the one by which the negative term decreases. More rigorously, consider the expression

$$k = \frac{1}{\cos(\theta)^2} - \frac{\sqrt{1-\alpha^2}}{\left(\alpha\sin(\theta)+\sqrt{1-\alpha^2}\cos(\theta)\right)\cos(\theta)}. \tag{6.34}$$

First, note that

$$\begin{aligned} k \geq 0 &\iff \sqrt{1-\alpha^2}\cos(\theta)^2 \leq \left(\alpha\sin(\theta)+\sqrt{1-\alpha^2}\cos(\theta)\right)\cos(\theta) && (6.35)\\ &\iff \cos(\theta) \leq \frac{\alpha}{\sqrt{1-\alpha^2}}\sin(\theta)+\cos(\theta) && (6.36)\\ &\iff 0 \leq \frac{\alpha}{\sqrt{1-\alpha^2}}\sin(\theta), && (6.37) \end{aligned}$$

which is always true when $0 \leq \theta \leq \pi/2$, which is always the case within the trigonometric domain that we consider. Then, if we denote the right hand side of (6.32) by $r_{n_0(w)}$, we have the recursive relation

$$r_{n_0(w)} = r_{n_0(w)+1}\frac{1}{\cos(\theta)^2} + k(\alpha\sin(\theta)+\sqrt{1-\alpha^2}\cos(\theta))^{n_0(w)}\left(\frac{\sqrt{1-\alpha^2}}{\cos(\theta)}\right)^{n-n_0(w)}.$$

We can see indeed that this defines an increasing sequence as we decrease $n_0(w)$, since the second summand is positive, and the first summand multiplies the previous value of $r$ by an amount greater than one. We have then successfully proved that (6.32) holds in the $\alpha \neq \frac{1}{\sqrt{2}}$ case, and therefore our candidate $Y$ is again feasible in the dual program (6.6) corresponding to the prover-verifier interaction we study.

□

In this proof, we prove dual feasibility by performing a direct sum decomposition of both the left-hand-side and right-hand-side matrices in the dual constraint for the dual program in (6.6). This allows us to analytically compute the value of the corresponding semidefinite program, while numerical approaches [9] using standard computational methods from the CVX [44] toolbox soon ($n = 4$ or $n = 5$) stop being able to analyze the problem under reasonable time constraints due to the size of the matrices involved. There are however promising ongoing efforts [97, 103] towards creating computational methods for the analysis of semidefinite programs that automate direct sum decompositions like the ones we perform here, through the study of symmetries in the corresponding matrices.

## 6.3 A motivation from mathematical finance for further study of quantum hedging

In [75], it was proved that quantum hedging cannot be used to increase the expected value of a prover-verifier interaction. That is to say, if we associate a value with each outcome of a prover-verifier interaction, it will always be optimal to play independently between parallel instances in order to maximize the total cumulative value. [1] In light of that bound, one might then wish to re-examine with a critical perspective the fictitious example for quantum hedging that we provide in Section 6.1.1, and ask why would any agent rationally decide to engage in quantum hedging behavior. It is possible to obtain an answer to this question by using the point of view of the logarithmic utility principle. Through this examination, one additionally obtains an answer to the question of what results concerning quantum hedging might be interesting to obtain given the previously mentioned work [54] that asymptotically bounds its magnitude enough to allow parallel repetition to work for the purposes of error reduction.

[1] This implies a fact that we will use later: If we have a prover-verifier interaction with a winning outcome and a losing outcome where the winning outcome can be obtained with certainty in at least one out of two parallel instances, then it must be possible to obtain the winning outcome with probability at least 0.5 when considering a single instance.

In this section, we apply then the framework of the logarithmic utility principle to the setting where a 2-round prover-verifier interaction (as described in 6.1) is conducted twice in parallel. The results imply that this application can strongly motivate further work on the limits to hedging, even if we know that it cannot be used to increase expected payoffs. This idea is exemplified by the result that we present in Claim 1, depict in Figure 6.2, and then prove in the rest of Section 6.3.2. This result shows that the range of incentive parameters where engaging in perfect quantum hedging is rational (as determined by the logarithmic utility principle) expands greatly in comparison with small changes in the advantage that the hedging provides. As earlier mentioned in Section 6.1.1, these considerations are not only of clear mathematical interest, but could also plausibly have a practical impact.

### 6.3.1 The logarithmic utility principle

The logarithmic utility of principle corresponds to a line of thought in finance [69] that believes that the utility associated with a given amount of wealth should scale linearly with the logarithm of the amount of wealth. One possible motivation for this elegant framework is that given a starting amount of wealth and a guaranteed periodic compounding rate, the amount of time it takes to reach a given wealth target through compounding scales linearly with the logarithm of the starting amount of wealth.

This framework has also the benefit of demonstrating how insurance-like transitions can be beneficial to both parties involved. In an introductory example of Bernoulli [21], one considers a merchant shipping goods with a value of 10,000 monetary units through a sea with a 5% chance of non-arrival. This merchant is offered insurance for the price of 800 units. In this context, if the merchant tries to maximize the logarithm of their wealth they will benefit from purchasing this insurance as long as their total wealth (excluding the goods) is below 5,043 units. Similarly, an insurer agent will benefit from offering this insurance as long as their wealth is above 14,243 units.

Finally, the logarithmic utility principle can also be used in the context of an investor who is compounding their capital through repeated investments, and who wishes to make choices about capital allocation for each of their possible next investment options, given their knowledge about expected returns. Through such an application, one derives the method that goes under the name of the Kelly criterion [68], which can be proven to be optimal in terms of optimizing long-term return rates, providing then another motivation for the consideration of the logarithmic utility principle itself.

### 6.3.2 Payoff regimes that encourage quantum hedging when two parallel instances of a protocol are conducted in parallel

We will examine here when do incentives lead to quantum hedging if a prover-verifier interaction is conducted twice in parallel, if an agent was to behave according to the logarithmic utility principle.

In particular, one can consider a prover-verifier interaction with a cost of participation equal to $s \in \mathbb{R}_+$ for the prover, and where a payoff of $c \in \mathbb{R}_+$ is obtained in case of obtaining the positive outcome of the interaction (i.e., payoffs equal to $-s$ and $c - s$ for the prover in the *no* and *yes* cases, respectively). Furthermore, we will let the initial wealth of the prover be denoted by $v \in \mathbb{R}_+$, and we will denote the wealth of the prover after paying the participation fee for two instances of the protocol as $v - 2s = x$. We will assume that $x > 0$, since the utility of non-positive wealth amounts is not well-defined in this approach.

Let the maximum probability of winning for the prover when a single repetition occurs be equal to $p < 1$. Furthermore, let us consider an arbitrary hedging (i.e. correlated) strategy for the prover in the case where two parallel instances are considered, with the corresponding probabilities of winning 0, 1 or 2 instances being given by $p_0$, $p_1$ and $p_2$, respectively. Then, if one uses the logarithmic utility principle, it will be better for the prover to engage in such a strategy rather than plan independently whenever it holds [2] that

$$\begin{aligned} &p_2 \log(x + 2d) + p_0 \log(x) + p_1 \log(x + d) \\ &> p^2 \log(x + 2d) + (1 - p)^2 \log(x) + 2p(1 - p) \log(x + d). \end{aligned} \tag{6.38}$$

One might reasonably also want to focus on situations where playing two instances of the game is worth it at all for the prover, again from the point of view of the logarithmic utility principle. If so, one obtains the additional condition that

$$p_2 \log(x + 2d) + p_0 \log(x) + p_1 \log(x + d) > \log(x + 2s). \tag{6.39}$$

#### Payoff regimes for perfect hedging

Once armed with equations (6.38) and (6.39), we can study which payoff regimes will incentivize known opportunities for quantum hedging. We can in particular do so in the

[2] Note that for convenience when taking derivatives, we will take our logarithms in base $e$.

cases where there is a non-trivial *perfect hedging* strategy (i.e. one where $p_1 = 1$, with $0.5 \leq p < 1$, with the lower bound on $p$ following from the results in [75] that we described at the beginning of Section 6.3.1). This study is motivated by the existence of such *perfect hedging* strategies. In particular, in the first example of quantum hedging in [77], we have such a strategy in a protocol where $p = \cos^2(\pi/8)$, and in [43], we have a strategy with $p_1 = 1$ in a coin-flipping setting where again $p = \cos^2(\pi/8)$.

Then, Equations (6.38) and (6.39) tell us that there will be an incentive to engage in perfect hedging rather than playing independently whenever it holds that

$$\log(x+d) > \; p^2 \log(x+2d) + (1-p)^2 \log(x) + 2p(1-p)\log(x+d) \tag{6.40}$$

$$\log(x+d) > \log(x+2s). \tag{6.41}$$

We can then ask for which values of $x$ and $d$ will this hold, as function of $p$, $x$ and $s$. It is clear that the Equation (6.41) imposes the constraint that

$$d > 2s. \tag{6.42}$$

As for equation (6.38), we can exponentiate each side, and obtain the equivalent condition that

$$x + d > (x+2d)^{p^2}(x+d)^{2p(1-p)}x^{(1-p)^2}. \tag{6.43}$$

If we let $r$ denote the ratio $d/x > 0$, we can write this as

$$x(1+r) > x^{p^2+2p(1-p)+(1-p)^2}(1+2r)p^2(1+r)^{2p(1-p)} \tag{6.44}$$

$$\iff \quad (1+r) > (1+2r)^{p^2}(1+r)^{2p(1-p)}. \tag{6.45}$$

While the dependence between $r$ and $p$ corresponding to Equation (6.45) is not clear at first sight, one can study the inequality and characterize analytically how large must $r$ be as a function of $p$, as stated in the following claim (its proof follows after our discussion of the consequences of the claim).

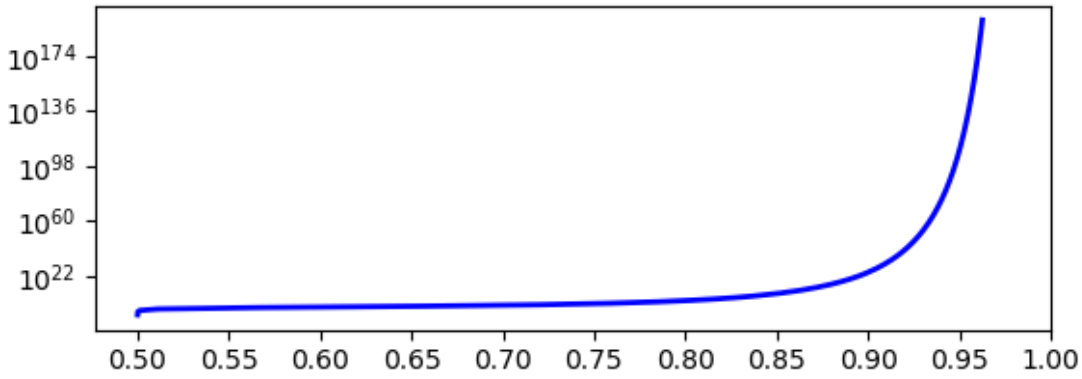


Figure 6.2: Below the curve, we have the region of the $(p, d/x)$ plane where perfect hedging in our 1-out-of-2 scenario is worth it under the logarithmic utility principle.

**Claim 1.** *In the setting under consideration here, there will be an incentive to engage in perfect hedging rather than playing independently whenever:*

$$d > 2s$$
$$d > x f^{-1}(p), \textit{where}:$$
$$f(0) = \frac{1}{2}, \textit{and for } x > 0$$
$$f(x) = g(x)\left(1 - \sqrt{1 - \frac{1}{g(x)}}\right)$$
$$g(x) = \frac{\log(1+x)}{\log\left(\frac{(1+x)^2}{1+2x}\right)}$$

If $p = \cos^2(\pi/8)$ (as in the previously mentioned examples), we derive then the constraint that $d > Cx$, where $C \approx 1.68 \times 10^{10}$. Furthermore, if one plots $f^{-1}(p)$, one obtains the graph that appears in Figure 6.2 (in logarithmic scale for the function's output).

We can see in the graph that as $p$ increases, the range of values of $d/x$ for which perfect hedging is worth it increases very quickly, even in a logarithmic plot. This graph clearly motivates then the search for a protocol where perfect hedging between two parallel instances is possible, and the optimal probability $p$ of winning a single instance is as small as possible, since that can increase greatly the range for the ratio $r = d/x$ for which the constraints here are satisfied.

Our intuition is that finding such a protocol with $p < \cos^2(\pi/8)$ should not be possible. Using the SDP formalism discussed in Section 6.1.2, this is equivalent to the following linear algebraic conjecture:

**Conjecture 2.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary finite-dimensional complex Hilbert spaces. Let $\rho \in \mathrm{D}(\mathcal{X})$ be an arbitrary density matrix operator. Let $Q_0 \in \mathrm{Pos}(\mathcal{Y} \otimes \mathcal{X})$, $Q_1 \in \mathrm{Pos}(\mathcal{Y} \otimes \mathcal{X})$, $X \in \mathrm{Pos}(\mathcal{Y} \otimes \mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{X})$, $Y \in \mathrm{Herm}(\mathcal{X})$ be such that*

$$\begin{aligned} Q_0 + Q_1 &= \mathcal{I}_{\mathcal{Y}} \otimes \rho \\ \langle X, Q_0 \otimes Q_0 \rangle &= 0 \\ \mathrm{Tr}_{\mathcal{Y} \otimes \mathcal{Y}}(X) &= \mathcal{I}_{\mathcal{X} \otimes \mathcal{X}} \\ \mathcal{I}_{\mathcal{Y}} \otimes Y &\geq Q_1. \end{aligned}$$

*Then, $Tr(Y) \geq \cos^2(\pi/8)$.*

Furthermore, we believe that it would not be possible to get past this barrier even if one was to increase the number of rounds allowed in the quantum prover-verifier interaction. Using the more algebraically complex formulation in [46, 47] for this $n$-round setting to represent this idea, one would then obtain a more algebraically complex extension of Conjecture 2.

We proceed now with the proof of Claim 1.

*Proof of Claim 1.* From Equation (6.45), it suffices to examine here the function $h(r, p) = (1 + r) - (1 + 2r)^{p^2}(1 + r)^{2p(1-p)}$ in the domain where $r \times p \in (0, +\infty) \times [0.5, 1)$. We seek to determine when is this function positive.

Observe that in our domain of concern, the function $h$ is continuous, since it is a composition of continuous functions (note that the function $exp(a, b) = a^b$ does not include $(0, 0)$ in its domain, but that is not an issue for us since the base in the exponentiations here is always $\geq 1$).

We can then first characterize the boundary within the $\mathbb{R}_+ \times [0.5, 1)$ subregion of the $(r, p)$ plane where $h(r, p) = 0$, and then examine the sign of the function $h$ in the regions in which this boundary divides the subregion. We have that:

$$
\begin{aligned}
& h(r,p) = 0 \\
\iff & (1+r) = (1+2r)^{p^2}(1+r)^{2p(1-p)} \\
\iff & \log(1+r) = p^2\log(1+2r) + 2p(1-p)\log(1+r) && \text{(taking logarithms)} \\
\iff & p^2(-\log(1+2r) + 2\log(1+r)) - 2p\log(1+r) + \log(1+r) = 0 \\
\iff & p^2 \log\left(\frac{(1+r)^2}{1+2r}\right) - 2p\log(1+r) + \log(1+r) = 0
\end{aligned} \tag{6.46}
$$

$$
\iff p = \frac{2\log(1+r) \pm \sqrt{4\log^2(1+r) - 4\log(1+r)\log\left(\frac{(1+r)^2}{1+2r}\right)}}{2\log\left(\frac{(1+r)^2}{1+2r}\right)} \quad \text{(by quadratic formula)}
$$

$$
\iff p = \frac{\log(1+r)}{\log\left(\frac{(1+r)^2}{1+2r}\right)}\left(1 \pm \sqrt{1 - \frac{\log\left(\frac{(1+r)^2}{1+2r}\right)}{\log(1+r)}}\right). \tag{6.47}
$$

We can now take the quadratic root corresponding to the minus-sign choice in Equation (6.47). This is because for the plus-sign choice, the corresponding value of the variable $p$ will be greater than 1, and not relevant to the boundary calculation we are concerned with here. One derives that by observing that since $r > 0$, $1 + 2r > 1 + r$, so $\frac{\log(1+r)}{\log\left(\frac{(1+r)^2}{1+2r}\right)} > 1$. Since $\sqrt{1-\frac{1}{x}} > 1 - \frac{1}{x}$ for $x > 1$, the value of $p$ must then be greater than 1.

After this choice of quadratic root, Equation (6.47) determines a function $f$ in the $(r,p)$ plane that associates a value of $p$ to every value of $r > 0$, given by

$$
\begin{aligned}
f(r) &= g(r)\left(1 - \sqrt{1 - \frac{1}{g(r)}}\right) \\
\text{where } g(r) &= \frac{\log(1+r)}{\log\left(\frac{(1+r)^2}{1+2r}\right)}.
\end{aligned} \tag{6.48}
$$

(note that our previous observation about the value of $\frac{\log(1+r)}{\log\left(\frac{(1+r)^2}{1+2r}\right)}$ gives us that $g(r) > 1$).

By the continuity of $h$, the sign of $h$ will then be the same for all values of $(r, p) \in (0, +\infty) \times [0.5, 1)$ where $p > f(r)$, and it will also be the same for all values of where $p < f(r)$. In order to prove that $h(r, p) < 0$ when $p > f(r)$ and $h(r, p) > 0$ when $p < f(r)$, one can compute then that for $r = 1$, $f(r) \approx 0.57$, $h(1, 0.60) \approx -0.071 < 0$, and $h(1, 0.55) \approx 0.035 > 0$.

We now prove that $f^{-1}$ exists and maps the interval $(0.5, 1)$ to $(0, +\infty)$. We will derive this by establishing that $f$ is an increasing monotone function that maps the interval $(0, +\infty)$ to $(0.5, 1)$.

In order to determine that $f$ is increasing, it is enough to determine that $g$ is decreasing. This is because we can write

$$f(r) = g(r) - \sqrt{g(r)^2 - g(r)}, \tag{6.49}$$

and therefore we can also write

$$\begin{aligned}
& f(r)' > 0 && (6.50)\\
\iff & g'(r) - \frac{2g(r)g'(r) - g(r)'}{2\sqrt{g(r)^2 - g(r)}} > 0 && (6.51)\\
\iff & 2g'(r) > \frac{2g(r)g'(r) - g(r)'}{\sqrt{g(r)^2 - g(r)}} && (6.52)\\
\iff & 2g'(r) > g'(r)\frac{2g(r) - 1}{\sqrt{g(r)^2 - g(r)}}. && (6.53)
\end{aligned}$$

We next check that

$$\begin{aligned}
& (2g'(r)\sqrt{g(r)^2 - g(r)})^2 - (g'(r)(2g(r) - 1))^2 && (6.54)\\
= & \, 4g'(r)^2(g(r)^2 - g(r)) - g'(r)^2(4(g(r)^2 - 4g(r) + 1) && (6.55)\\
= & \, -g'(r)^2 && (6.56)\\
< & \, 0. && (6.57)
\end{aligned}$$

Therefore, $|2g'(r)| < |g'(r)\frac{2g(r)-1}{\sqrt{g(r)^2-g(r)}}|$. Since $g(r) > 1$, this means that the inequality in (6.53) will hold if and only if $g'(r) < 0$.

We have then proved that if $g$ is decreasing, $f$ is increasing. We now check that $g(r)$ is indeed decreasing at all values of $r \in (0, +\infty)$. In order to do so, we compute

$$g'(r) = \frac{(1+2r)\log((1+r)^2/(1+2r)) - 2r\log(1+r)}{(1+r)(1+2r)\log^2((1+r)^2/(1+r))}. \tag{6.58}$$

It is clear from (6.58) that $g'(r)$ will be negative if and only if numerator is negative. We denote this numerator as $g'_{num}(r)$. When we examine its sign, we find that

$$\begin{aligned} & g'_{num}(r) < 0 && (6.59)\\ \iff & (1+2r)(2\log(1+r) - \log(1+2r)) - 2r\log(1+r) < 0 && (6.60)\\ \iff & 2(1+r)\log(1+r) - (1+2r)\log(1+2r) < 0. && (6.61) \end{aligned}$$

In order to verify that the inequality in (6.61) holds for all values of $r \in (0, +\infty)$, it is enough to establish that

$$\lim_{r\to 0^+} 2(1+r)\log(1+r) - (1+2r)\log(1+2r) = 0 \tag{6.62}$$

and

$$\begin{aligned} \frac{d}{dr}(2(1+r)\log(1+r) - (1+2r)\log(1+2r)) &= 2(\log(1+r) - \log(1+2r)) && (6.63)\\ &= 2\log\left(\frac{1+r}{1+2r}\right) && (6.64)\\ &< 0. && (6.65) \end{aligned}$$

Finally, we have by L'Hospital's rule that:

$$\lim_{r\to\infty} g(r) = \lim_{r\to\infty} \frac{\frac{d}{dr}\log(1+r)}{\frac{d}{dr}\log\left(\frac{(1+r)^2}{1+2r}\right)} = \lim_{r\to\infty} \frac{\frac{1}{1+r}}{\frac{2r}{2r^2+3r+1}} = 1 \tag{6.66}$$

$$\lim_{r\to 0^+} g(r) = \lim_{r\to 0^+} \frac{\frac{d}{dr}\log(1+r)}{\frac{d}{dr}\log\left(\frac{(1+r)^2}{1+2r}\right)} = \lim_{r\to 0^+} \frac{\frac{1}{1+r}}{\frac{2r}{2r^2+3r+1}} = +\infty \tag{6.67}$$

$$\lim_{r\to 0^+} f(r) = \lim_{r\to 0^+} \frac{\frac{d}{dr}\left(1-\sqrt{1-\frac{1}{g(r)}}\right)}{\frac{d}{dr}\frac{1}{g(r)}} = \lim_{r\to 0^+} \frac{1}{2\sqrt{1-\frac{1}{g(r)^2}}} = \frac{1}{2}. \tag{6.68}$$

Equations (6.66)-(6.68) imply then that $\lim_{r\to\infty} f(r) = 1$ and $\lim_{r\to 0^+} f(r) = 1/2$, so we have that $f$ is surjective into the interval $(0.5, 1)$, as desired.

The last step is to verify that when $p = \frac{1}{2}$, it is true for any value of $r > 0$ that $h(r, 1/2) > 0$. We have indeed that:

$$(1+r) - (1+2r)^{1/4}(1+r)^{1/2} > 0 \tag{6.69}$$

$$\iff (1+r)^{1/2} > (1+2r)^{1/4} \tag{6.70}$$

$$\iff (1+r)^2 > (1+2r) \tag{6.71}$$

$$\iff r^2 > 0. \tag{6.72}$$

We arrive then to the region identified in the statement of Claim 1.

□

### Other models and questions

The analysis here serves as a proof-of-concept for the application of the logarithmic utility principle to quantum hedging situations, where we have limited ourselves to standard quantum prover-verifier applications with two parallel copies of an interaction. One could aim to repeat this analysis then (analytically or numerically) in more complex cases where $n$ instances of a protocol are conducted in parallel, or even seek its application in variations of our prover-verifier setting (e.g. with multiple cooperating or competing provers).

In order to obtain an answer to Conjecture 2, and more generally obtain a better understanding of quantum hedging, it might be of interest to consider the object of all correlation patterns that can arise from quantum-hedging. For example, one can consider the object of all tuples $(p_0, p_1, p_2)$ indicating the probability of 0, 1 and 2 wins after a given strategy is used when playing two parallel instances of a prover-verifier interaction, and aim for tight bounds on this object over all prover-verifier interactions as a function of the optimal probability $p$ of winning a single instance and the optimal probability $q$ of achieving the losing outcome in a single instance.

# References


[1] Oblivious Turing machine emulation lower bound. Theoretical Computer Science Stack Exchange. URL:https://cstheory.stackexchange.com/q/10645 (version: 2019-06-09).

[2] Leonard M Adleman, Jonathan DeMarrais, and Ming-Deh A Huang. Quantum computability. *SIAM Journal on Computing*, 26(5):1524–1540, 1997.

[3] Dorit Aharonov, Alexei Kitaev, and Noam Nisan. Quantum circuits with mixed states. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 20–30, 1998.

[4] Chris Aholt, Sameer Agarwal, and Rekha Thomas. A QCQP approach to triangulation. In *European Conference on Computer Vision*, pages 654–667. Springer, 2012.

[5] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[6] Juan Miguel Arrazola, Petros Wallden, and Erika Andersson. Multiparty quantum signature schemes. *Quantum Information & Computation*, 16(5-6):435–464, 2016.

[7] Pablo Arrighi, Simon Martiel, and Simon Perdrix. Reversible causal graph dynamics: invertibility, block representation, vertex-preservation. *Natural Computing*, pages 1–22, 2019.

[8] Pablo Arrighi, Vincent Nesme, and Reinhard Werner. Unitarity plus causality implies localizability. *J. Comput. Syst. Sci.*, 77(2), March 2011.

[9] Srinivasan Arunachalam, Abel Molina, and Vincent Russo. Semidefinite programs for quantum hedging framework. https://bitbucket.org/vprusso/quantum-hedging/src/master/Code/, 2016.

[10] Srinivasan Arunachalam, Abel Molina, and Vincent Russo. Quantum hedging in two-round prover-verifier interactions. In *12th Conference on the Theory of Quantum Computation, Communication and Cryptography*, 2018.

[11] Koenraad MR Audenaert and Stefan Scheel. Quantum tomographic reconstruction with error bars: a Kalman filter approach. *New Journal of Physics*, 11(2):023028, 2009.

[12] Somshubhro Bandyopadhyay, Rahul Jain, Jonathan Oppenheim, and Christopher Perry. Conclusive exclusion of quantum states. *Physical Review A*, 89(2):022336, 2014.

[13] Adriano Barenco, Charles Bennett, Richard Cleve, David DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52:3457–3467, 1995.

[14] Jonathan Barrett, Eric G Cavalcanti, Raymond Lal, and Owen JE Maroney. No $\psi$-epistemic model can fully explain the indistinguishability of quantum states. *Physical Review Letters*, 112(25):250403, 2014.

[15] Ingemar Bengtsson and Karol Zyczkowski. *Geometry of Quantum States: An Introduction to Quantum Entanglement*. Cambridge University Press, 2007.

[16] Charles H Bennett. Logical reversibility of computation. *IBM journal of Research and Development*, 17(6):525–532, 1973.

[17] Charles H Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, 1989.

[18] Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[19] Cedric Beny. Causal structure of the entanglement renormalization ansatz. *New Journal of Physics*, 15(2):023020, 2013.

[20] Michael R Beran and Scott M Cohen. Nonoptimality of unitary encoding with quantum channels assisted by entanglement. *Physical Review A*, 78(6):062337, 2008.

[21] Daniel Bernoulli. Exposition of a new theory on the measurement of risk (translation to english). *Econometrica*, 22(1):23–36, 1954.

[22] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing, ACM*, pages 11–20, 1993.

[23] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.

[24] Sergio Boixo, Vadim N Smelyanskiy, Alireza Shabani, Sergei V Isakov, Mark Dykman, Vasil S Denchev, Mohammad H Amin, Anatoly Yu Smirnov, Masoud Mohseni, and Hartmut Neven. Computational multiqubit tunnelling in programmable quantum annealers. *Nature Communications*, 7:10327, 2016.

[25] Subhonmesh Bose, Dennice F Gayme, K Mani Chandy, and Steven H Low. Quadratically constrained quadratic programs on acyclic graphs with application to power flow. *IEEE Transactions on Control of Network Systems*, 2(3):278–287, 2015.

[26] Hans J Briegel, David E Browne, Wolfgang Dür, Robert Raussendorf, and Maarten Van den Nest. Measurement-based quantum computation. *Nature Physics*, 5(1):19–26, 2009.

[27] Todd A Brun, Min-Hsiu Hsieh, and Christopher Perry. Compatibility of state assignments and pooling of information. *Physical Review A*, 92(1):012107, 2015.

[28] Harry Buhrman, John Tromp, and Paul Vitányi. Time and space bounds for reversible simulation. In *International Colloquium on Automata, Languages, and Programming*, pages 1017–1027. Springer, 2001.

[29] Marco Carpentieri. On the simulation of quantum Turing machines. *Theoretical Computer Science*, 304(1-3):103–128, 2003.

[30] Carlton M Caves, Christopher A Fuchs, and Rüdiger Schack. Conditions for compatibility of quantum-state assignments. *Physical Review A*, 66(6):062111, 2002.

[31] Andrew M Childs. Lecture notes on quantum algorithms. 2017. Available at https://www.cs.umd.edu/~amchilds/qa/qa.pdf.

[32] John F Clauser, Michael A Horne, Abner Shimony, and Richard A Holt. Proposed experiment to test local hidden-variable theories. *Physical Review Letters*, 23:880–884, 1969.

[33] John Conway. *A Course in Operator Theory*. American Mathematical Society, 2000.

[34] Christopher M Dawson and Michael A Nielsen. The Solovay-Kitaev algorithm. *Quantum Information & Computation*, 6(1):81–95, 2006.

[35] David Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 400(1818):97–117, 1985.

[36] Yonina C Eldar, Alexandre Megretski, and George C Verghese. Designing optimal quantum detectors via semidefinite programming. *IEEE Transactions on Information Theory*, 49(4):1007–1012, 2003.

[37] Yonina C Eldar, Mihailo Stojnic, and Babak Hassibi. Optimal quantum detectors for unambiguous detection of mixed states. *Physical Review A*, 69(6):062318, 2004.

[38] Youping Fan and Bernd Tibken. Optimization problems of determining the C-numerical range. *IFAC Proceedings Volumes*, 41(2):10051–10056, 2008.

[39] Bill Fefferman and Cedric Yen-Yu Lin. A complete characterization of unitary quantum space. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, 2018.

[40] Uriel Feige. On the success probability of the two provers in one-round proof systems. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference*, pages 116–123. IEEE, 1991.

[41] Jaromír Fiurášek and Miroslav Ježek. Optimal discrimination of mixed quantum states involving inconclusive results. *Physical Review A*, 67(1):012321, 2003.

[42] Michael Freedman and Matthew B Hastings. Classification of quantum cellular automata. 2018.

[43] Maor Ganz and Or Sattath. Quantum coin hedging, and a counter measure. In *12th Conference on the Theory of Quantum Computation, Communication and Cryptography*, 2018.

[44] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1, 2014.

[45] Greg N Gregoriou. *Funds of hedge funds: performance, assessment, diversification, and statistical properties*. Elsevier, 2011.

[46] Gus Gutoski and John Watrous. Toward a general theory of quantum games. In *Proceedings of the 39th annual ACM symposium on Theory of computing*, pages 565–574. ACM, 2007.

[47] Gustav Gutoski. *Quantum Strategies and Local Operations*. PhD thesis, University of Waterloo, 2010.

[48] Erkka Haapasalo, Teiko Heinosaari, and Juha-Pekka Pellonpää. Quantum measurements on finite dimensional systems: relabeling and mixing. *Quantum Information Processing*, 11(6):1751–1763, 2012.

[49] Vojtěch Havlíček and Jonathan Barrett. Simple communication complexity separation from quantum state antidistinguishability. *Physical Review Research*, 2(1):013326, 2020.

[50] Patrick Hayden, Kevin Milner, and Mark M Wilde. Two-message quantum interactive proofs and the quantum separability problem. *Quantum Information & Computation*, 14(5&6):384–416, 2014.

[51] Teiko Heinosaari and Oskari Kerppo. Antidistinguishability of pure quantum states. *Journal of Physics A: Mathematical and Theoretical*, 51(36):365303, 2018.

[52] Bas Hensen, Hannes Bernien, Anaïs E Dréau, Andreas Reiserer, Norbert Kalb, Machiel S Blok, Just Ruitenberg, Raymond FL Vermeulen, Raymond N Schouten, Carlos Abellán, et al. Loophole-free Bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526(7575):682, 2015.

[53] Thomas Holenstein. Parallel repetition: Simplification and the no-signaling case. *Theory of Computing*, 5(1):141–172, 2009.

[54] Taylor Hornby. Concentration bounds from parallel repetition theorems. Master's thesis, University of Waterloo, 2018.

[55] Yongwei Huang and Daniel P Palomar. Randomized algorithms for optimal solutions of double-sided QCQP with applications in signal processing. *IEEE Transactions on Signal Processing*, 62(5):1093–1108, 2014.

[56] Rahul Jain, Sarvagya Upadhyay, and John Watrous. Two-message quantum interactive proofs are in PSPACE. In *2009 IEEE 50th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 534–543. IEEE, 2009.

[57] Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. MIP*= RE. *arXiv preprint arXiv:2001.04383*, 2020.

[58] Cédric Josz and Daniel K Molzahn. Moment/sum-of-squares hierarchy for complex polynomial optimization. *arXiv preprint 1508.02068*, 2015.

[59] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An introduction to quantum computing*. Oxford university press, 2007.

[60] Alexei Kitaev, Alexander Shen, and Mikhail Vyalyi. *Classical and quantum computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, 2002.

[61] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. Fast and efficient exact synthesis of single-qubit unitaries generated by clifford and t gates. *Quantum Information & Computation*, 13(7-8):607–630, 2013.

[62] Aritra Konar and Nicholas D Sidiropoulos. Hidden convexity in QCQP with Toeplitz-Hermitian quadratics. *IEEE Signal Processing Letters*, 22(10):1623–1627, 2015.

[63] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 839–858. IEEE, 2016.

[64] Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible space equals deterministic space. *Journal of Computer and System Sciences*, 60(2):354–367, 2000.

[65] Quanzhong Li, Qi Zhang, and Jiayin Qin. A special class of fractional QCQP and its applications on cognitive collaborative beamforming. *IEEE Transactions on Signal Processing*, 62(8):2151–2164, 2014.

[66] Yeong-Cherng Liang and Andrew C Doherty. Bounds on quantum correlations in Bell-inequality experiments. *Physical Review A*, 75(4):042103, 2007.

[67] Zi-Wen Liu, Christopher Perry, Yechao Zhu, Dax Enshan Koh, and Scott Aaronson. Doubly infinite separation of quantum information and communication. *Physical Review A*, 93(1):012347, 2016.

[68] Leonard C MacLean, Edward O Thorp, and William T Ziemba. Good and bad properties of the Kelly criterion. *Risk*, 20(2):1, 2010.

[69] Leonard C MacLean, Edward O Thorp, and William T Ziemba. *The Kelly capital growth investment criterion: Theory and practice*, volume 3. World Scientific, 2011.

[70] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267. IEEE, 2018.

[71] Simon Martiel. *Algorithmical and mathematical approaches of causal graph dynamics*. PhD thesis, Université Nice Sophia Antipolis, 2015.

[72] N David Mermin. Simple unified form for the major no-hidden-variables theorems. *Physical Review Letters*, 65:3373–3376, 1990.

[73] N David Mermin. Whose knowledge? In *Quantum [Un] speakables*, pages 271–280. Springer, 2002.

[74] Andrew Miller and Iddo Bentov. Zero-collateral lotteries in bitcoin and ethereum. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 4–13. IEEE, 2017.

[75] Abel Molina. Parallel repetition of prover-verifier quantum interactions. Master's thesis, University of Waterloo, 2012.

[76] Abel Molina. POVMs are equivalent to projections for perfect state exclusion of three pure states in three dimensions. *Quantum*, 3:117, 2019.

[77] Abel Molina and John Watrous. Hedging bets with correlated quantum strategies. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 468(2145):2614–2629, 2012.

[78] Abel Molina and John Watrous. Revisiting the simulation of quantum Turing machines by quantum circuits. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 475(2226), 2019.

[79] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2002.

[80] Harumichi Nishimura and Masanao Ozawa. Quantum oracles and computational complexity. *Departmental Bulletin Paper- Research Institute for Mathematical Sciences, Kyoto University*, 2000.

[81] Harumichi Nishimura and Masanao Ozawa. Computational complexity of uniform quantum circuit families and quantum Turing machines. *Theoretical Computer Science*, 276(1-2):147–181, 2002.

[82] Masanao Ozawa and Harumichi Nishimura. Local transition functions of quantum Turing machines. *RAIRO-Theoretical Informatics and Applications*, 34(5):379–402, 2000.

[83] Jaehyun Park and Stephen Boyd. General heuristics for nonconvex quadratically constrained quadratic programming. *arXiv preprint 1703.07870*, 2017.

[84] Kalyanapuram R Parthasarathy. Extremal decision rules in quantum hypothesis testing. *Infinite Dimensional Analysis, Quantum Probability and Related Topics*, 2(04):557–568, 1999.

[85] Rudolf Ernst Peierls. *More Surprises in Theoretical Physics*, volume 19. Princeton University Press, 1991.

[86] Simon Perdrix. Towards Observable Quantum Turing Machines: Fundamentals, Computational Power, and Universality. *International Journal of Unconventional Computing*, 7(4):291–311, 2011.

[87] Simon Perdrix and Philippe Jorrand. Classically controlled quantum computation. *Mathematical Structures in Computer Science*, 16(4):601–620, 2006.

[88] Asher Peres. Incompatible results of quantum measurements. *Physics Letters A*, 151:107–108, 1990.

[89] Christopher Perry. *Conclusive exclusion of quantum states and aspects of thermomajorization*. PhD thesis, UCL (University College London), 2016.

[90] Christopher Perry, Rahul Jain, and Jonathan Oppenheim. Communication tasks with infinite quantum-classical separation. *Physical Review Letters*, 115(3):030504, 2015.

[91] Nicholas Pippenger and Michael Fischer. Relations among complexity measures. *Journal of the ACM*, 26(2):361–381, 1979.

[92] Maximilian Puelma Touzel, Rob Adamson, and Aephraim Steinberg. Optimal bounded-error strategies for projective measurements in nonorthogonal-state discrimination. *Physical Review A*, 76(6):062314, 2007.

[93] Matthew F Pusey, Jonathan Barrett, and Terry Rudolph. On the reality of the quantum state. *Nature Physics*, 8(6):475–478, 2012.

[94] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

[95] Ran Raz. Quantum information and the PCP theorem. In *2005 IEEE 46th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 459–468. IEEE, 2005.

[96] Ran Raz and Avishay Tal. Oracle separation of BQP and PH. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, pages 13–23, 2019.

[97] Denis Rosset. Symdpoly: symmetry-adapted moment relaxations for noncommutative polynomial optimization. *arXiv preprint arXiv:1808.09598*, 2018.

[98] John Savage. Computational work and time on finite machines. *Journal of the ACM*, 19(4):660–674, 1972.

[99] Gael Sentís, Bernat Gendra, Stephen D Bartlett, and Andrew C Doherty. Decomposition of any quantum measurement into extremals. *Journal of Physics A: Mathematical and Theoretical*, 46(37):375302, 2013.

[100] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 2012.

[101] Blake C Stacey. SIC-POVMs and compatibility among quantum states. *Mathematics*, 4(2):36, 2016.

[102] Masamichi Takesaki. *Theory of Operator Algebras II*, volume 125 of *Encyclopaedia of Mathematical Sciences*. Springer, 2013.

[103] Armin Tavakoli, Denis Rosset, and Marc-Olivier Renou. Enabling computation of correlation bounds for finite-dimensional quantum systems via symmetrization. *Physical review letters*, 122(7):070501, 2019.

[104] Guo Chuan Thiang. Some attempts at proving the non-existence of a full set of mutually unbiased bases in dimension 6. *arXiv preprint 1012.3147*, 2010.

[105] Tommaso Toffoli. Reversible computing. In *International Colloquium on Automata, Languages, and Programming*, pages 632–644. Springer, 1980.

[106] Alan Mathison Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(1):230–265, 1937.

[107] Dan Ventura and Tony Martinez. Quantum harmonic sieve: Learning dnf with a classical example oracle. *arXiv preprint quant-ph/9805043*, 1998.

[108] Támas Vértesi and Erika Bene. Two-qubit Bell inequality for which positive operator-valued measurements are relevant. *Physical Review A*, 82(6):062115, 2010.

[109] Dong-Sheng Wang. A local model of quantum Turing machines. *Quantum Information & Computation*, 20(3-4):213–229, 2020.

[110] Qisheng Wang and Mingsheng Ying. Quantum random access stored-program machines. *arXiv preprint arXiv:2003.03514*, 2020.

[111] John Watrous. Personal Communication.

[112] John Watrous. On one-dimensional quantum cellular automata. In *Proceedings of the IEEE 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 528–537. IEEE, 1995.

[113] John Watrous. Quantum computational complexity. *Encyclopedia of complexity and systems science*, pages 7174–7201, 2009.

[114] John Watrous. *The theory of quantum information*. Cambridge University Press, 2018.

[115] Stephanie Wehner. Entanglement in interactive proof systems with binary answers. In *STACS 2006*, pages 162–171. Springer, 2006.

[116] Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412):eaam9288, 2018.

[117] Graeme Weir, Stephen M. Barnett, and Sarah Croke. Optimal discrimination of single-qubit mixed states. *Physical Review A*, 96(2):022312, 2017.

[118] Mark M Wilde. *Quantum information theory*. Cambridge University Press, 2013.

[119] Andrew Yao. Quantum circuit complexity. In *Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science*, pages 352–361, 1993.

[120] Huangjun Zhu, Yong Siah Teo, and Berthold-Georg Englert. Two-qubit symmetric informationally complete positive-operator-valued measures. *Physical Review A*, 82(4):042308, 2010.