RAPHAEL DIAS DA SILVA

Translating one-way quantum computation to the circuit model: methods and applications

Niterói

September 2013

RAPHAEL DIAS DA SILVA

Translating one-way quantum computation to the circuit model: methods and applications

Tese apresentada ao Curso de Pós-Graduação em Física da Universidade Federal Fluminense, como requisito parcial para obtenção do Título de Doutor em Física.

Orientador:
Prof. Dr. Ernesto Fagundes Galvão

Universidade Federal Fluminense
Instituto de Física

Niterói

September 2013

S586t Silva, Raphael Dias da.

Translating one-way quantum computation to the circuit model: methods and applications / Raphael Dias da Silva ; orientador: Ernesto F. Galvão. -- Niterói, 2013.
166 f. : il.

Tese (Doutorado) – Universidade Federal Fluminense, Instituto de Física,2013.
Bibliografia: f. 159-166.

1.COMPUTAÇÃO QUÂNTICA. 2.CIRCUITO QUÂNTICO. I.Galvão, Ernesto Fagundes, Orientador. II.Universidade Federal Fluminense. Instituto de Física, Instituição responsável. III.Título.

CDD 530.120285

INSTITUTO DE FÍSICA
Universidade Federal Fluminense
CURSO DE PÓS-GRADUAÇÃO EM FÍSICA
RUA GAL MILTON TAVARES DE SOUZA, SN
24210-346 – NITERÓI - RIO DE JANEIRO
TEL: (21)2629-5878 - FAX: 2629-5887
E-MAIL: cpg@ if.uff.br

Ata dos trabalhos finais da Comissão Examinadora da tese apresentada por ***Raphael Dias da Silva.*** *No segundo dia do mês de setembro de dois mil e treze, às quatorze horas, reuniram-se no Instituto de Física da Universidade Federal Fluminense os membros da Comissão Examinadora constituída pelos professores doutores Ernesto Fagundes Galvão (IF/UFF), Marcelo Silva Sarandy (IF/UFF), Fernando da Rocha Vaz Bandeira de Melo (CBPF), Roberto Imbuzeiro Moraes Felinto de Oliveira (IMPA) e Marcelo de Oliveira Terra Cunha (UFMG), sob a presidência do primeiro, para prova pública de apresentação da tese intitulada* ***"Translating one-way quantum computation to the circuit model: methods and apppplications",*** *tendo em vista as exigências do Regulamento Específico do curso de Física relacionadas com a conclusão do Doutorado em Física pela Universidade Federal Fluminense. A tese foi elaborada sob a orientação do professor Ernesto Fagundes Galvão. Após a exposição do trabalho, o aluno respondeu às questões formuladas pelos integrantes da Comissão Examinadora, que apresentou parecer no sentido de aprová-lo. Para constar, foi lavrada a presente ata, que vai assinada pelos membros da Comissão Examinadora e pelo doutorando.*

Niterói, dois de setembro de dois mil e treze.

Dr. Ernesto Fagundes Galvão _____
Dr. Marcelo Silva Sarandy _____
Dr. Fernando da Rocha Vaz Bandeira de Melo _____
Dr. Roberto Imbuzeiro M.Felinto de Oliveira _____
Dr. Marcelo de Oliveira Terra Cunha _____
Raphael Dias da Silva _____

À minha mãe e à minha avó, os pilares da minha vida.

"All that is sacred comes from youth
Dedication, naive and true."
Not For You, Pearl Jam.

Agradecimentos

Eu gostaria de agradecer ao meu orientador Ernesto F. Galvão por dedicar tanto do seu tempo à minha orientação desde a iniciação científica. Agradeço pelas inúmeras discussões que tivemos e por suas tentativas de me explicar conceitos físicos com o máximo de clareza possível. Sua atenção aos detalhes e zelo pela precisão são responsáveis pela qualidade que esta tese possui. Todas as imperfeições, no entanto, são por minha conta. Ernesto, obrigado pela sua paciência e dedicação por todos esses anos!

Gostaria também de agradecer à Elham Kashefi, que sempre se mostrou incrivelmente entusiasmada com minha pesquisa desde o início do meu doutorado. Muito foi aprendido quando estive sob sua tutela durante meu estágio de doutorado em Edimburgo, Escócia. A convivência com ela foi muito importante para a minha formação e na decisão de seguir a carreira acadêmica.

Durante o meu estágio de doutorado no exterior eu pude conhecer várias pessoas especiais. Agradeço aos meus amigos Einar Pius e Vedran Dunjko por terem me recebido tão bem em Edimburgo. Obrigado por todas as discussões que tivemos, pelas conversas sobre a vida e pelas cervejas. Eu aprendi muito com a convivência com vocês dois!

Várias outras pessoas contribuíram para que minha estadia na nublada Edimburgo fosse mais prazerosa. Dentre elas eu gostaria de agradecer especialmente ao Leonardo Disilvestro, Tomoyuke Morimae, Riinu Ots, Vincent Danos e à Stefania Bonà.

Gostaria também de agradecer ao Damian Markham pelo convite para apresentar meu trabalho em Paris e por me receber tão bem durante o período em que visitei seu grupo de pesquisa. Gostaria também de agradecer ao Simon Perdrix e ao Mehdi Mhalla pelo interesse em meu trabalho e pelo convite de visitar seu grupo de pesquisa em Grenoble, França. As discussões que tive com vocês foram muito esclarecedoras e agradáveis.

Gostaria também de agradecer aos meus amigos do IF-UFF que em muito contribuíram para a minha formação como físico e como pessoa. Uma lista incompleta é formada por Igor Diniz, Carol Vannier, Cadu, Daniel Brod, Dayanne Amaral, Wanessa Sarzêdas, Alexandre Lima, Thiago Caramês e Cleidson Castro, aos quais agradeço fortemente. Vários professores do IF-UFF tiveram um papel importantíssimo na minha for-

mação acadêmica e também como ser humano, dentre eles gostaria de agradecer especialmente ao Marco Moriconi, Daniel Jonathan, Antônio Zelaquett, Nivaldo Lemos e Antonio Costa.

Agradeço também aos meus amigos de mais de uma década Bruno Nazar, Karlos Dyego e Karla Louise. O incentivo e a amizade de vocês têm sido determinante na minha formação como ser humano.

Gostaria especialmente de agradecer à minha mãe e à minha avó pelo amor incondicional e pela crença que sempre depositaram em mim. Muito obrigado por me darem a oportunidade de realizar meus sonhos e por nunca permitir que as dificuldades de nossa família chegassem até mim. Absolutamente tudo que sou e algum dia serei eu devo a vocês duas. Eu amo vocês!

Por fim agradeço à minha namorada Helena, pelo apoio incondicional que sempre me deu. Helena, meu amor, obrigado por acreditar em mim nos momentos em que nem eu acreditava. Sem você ao meu lado tudo teria sido muito mais difícil. Te amo demais!

Acknowledgements

I would like to thank my advisor Ernesto F. Galvão for all the time he has dedicated to mentoring me, ever since I was an undergrad. I thank him for the numerous discussions, and for his efforts in transmitting physical concepts in a crystal clear manner. His attention to detail and precision were essential for the quality of this work. All its imperfections, however, are my own responsibility. Ernesto, thank you for your patience and dedication through all these years!

I would also like to thank Elham Kashefi, who has shown such great enthusiasm in my research since the beginning of my PhD. I learned a great deal under her mentoring during my stay in Edinburgh, Scotland. Working with her greatly contributed both to my academic experience and my decision to pursue this career.

During my stay abroad I was also fortunate to meet a number of special people. I thank my friends Einar Pius and Vedran Dunjko for receiving me so warmly in Edinburgh. Thank you for the discussions we had, for the great talks, and for the beers. I learned a great deal from you guys!

Many others contributed to brighten my stay in rainy Edinburgh. Among them, I would like to particularly thank Leonardo Disilvestro, Tomoyuke Morimae, Riinu Ots, Vincent Danos, and Stefania Bonà.

I would also like to thank Damian Markham for inviting me to present my research in Paris, and for receiving me so warmly during my visit to his research group. I would also like to thank Simon Perdrix and Mehdi Mhalla for the interest shown in my work, and for inviting me to visit their research group in Grenoble, France. The talks we had were both pleasant and enlightening.

I also thank my friends from the Physics Institute at UFF, who greatly contributed for both my academic and personal growth. An (incomplete) list is Igor Diniz, Carol Vannier, Cadu, Daniel Brod, Dayanne Amaral, Wanessa Sarzêdas, Alexandre Lima, Thiago Caramês, and Cleidson Castro, all to whom I'm deeply grateful. I am also grateful to the many teachers who played a major role in my development as a scientist and human being, particularly Marco Moriconi, Daniel Jonathan, Antônio Zelaquett, Nivaldo Lemos,

and Antonio Costa.

I also thank my longtime friends Bruno Nazar, Karlos Dyego and Karla Louise. Your friendship and support have immensely contributed to the person I am today.

I would especially like to thank my mother and my grandmother for their unconditional love, and for always believing in me. Thank you for providing me with the opportunity to follow my dreams, and for never letting our family's hardships reach me. Absolutely everything that I am and will ever be I owe it to you. I love you!

Finally, I would like to thank my girlfriend Helena for the ever-present unconditional support. Helena, my love, thank you for believing in me even when I didn't believe in myself. Everything would have been unbearably harder without you by my side. I love you dearly!

Abstract

In this thesis I study the one-way quantum computation (1WQC) model and some applications of the different ways of translating 1WQC algorithms into the circuit model. In a series of recent results, different sets of conditions for implementing a computation deterministically in the one-way model have been proposed, each of them with their own properties. Some of those sets of conditions - generically known as flow conditions - try to explore the distinct parallel power of the 1WQC model, by increasing the number of operations that can be performed simultaneously. Here I contribute to this line of research by defining a new type of flow, which I call the signal-shifted flow (SSF), which has an interesting parallel structure that equals that of a depth-optimal flow.

I also introduce a new framework for translating 1WQC algorithms into the circuit model. This translation preserves not only the computation performed but also some features of the 1WQC algorithm design. Within this framework I give two algorithms, each implementing a different translation procedure: the first gives compact (in space use) circuits for Regular Flow one-way computations, and the second does the same for SSF one-way computations. As an application of the SSF translation procedure, I combine it with other translation and optimization techniques to give an automated quantum circuit optimization procedure. This procedure is based on back-and-forth translation between the 1WQC and the circuit model, using 1WQC techniques to time-optimize computations in the circuit model.

In the second part of this thesis, I use 1WQC tools to analyze quantum circuits interacting with closed timelike curves (CTCs). I do so by translating to the 1WQC model CTC-assisted circuits, and then showing that in some cases they can be shown to be equivalent to time-respecting circuits. The predictions obtained in those cases are exactly those of the quantum CTC model based on post-selected teleportation, proposed by Bennett, Schumacher and Svetlichny (BSS). This enabled us to show that the BSS model for quantum CTCs makes predictions which disagree with those of the highly influential CTC model proposed by David Deutsch.

Resumo

Nesta tese eu estudo o modelo de computação quântica baseada em medições (CQBM) e algumas aplicações das diferentes maneiras de traduzir algoritmos de CQBM para o modelo de circuitos. Em uma série de resultados recentes, vários conjuntos de condições para implementar uma computação deterministicamente no modelo de CQBM têm sido propostas, cada um deles com diferentes propriedades. Alguns desses conjuntos de condições - genericamente conhecidos como condições de fluxo (*flow*) - tentam explorar o poder de paralelização do modelo de CQBM, aumentando o número de operações que podem ser realizadas simultaneamente. Aqui eu contribuo para essa linha de pesquisa definindo um novo tipo de fluxo, chamado fluxo de sinal deslocado (FSD), que tem uma estrutura paralela interessante que se iguala ao de um fluxo ótimo, do ponto de vista temporal.

Eu também introduzo um novo sistema para traduzir algoritmos de CQBM para o modelo de circuitos. Esta tradução preserva não só a computação, mas também outras características de algoritmos em CQBM. Usando esse sistema eu desenvolvo dois algoritmos, cada um capaz de executar um procedimento de tradução diferente: o primeiro obtém circuitos compactos a partir de computações com fluxo regular, e o segundo faz o mesmo para computações com FSD. Como uma aplicação do procedimento de tradução de computações com FSD, eu combino esse procedimento com outras técnicas de tradução e otimização para desenvolver um procedimento automático de otimização de circuitos quânticos. Esse procedimento é baseado em traduções nos dois sentidos entre os modelos de CQBM e de circuitos, usando técnicas de CQBM para otimizar circuitos quânticos

Na segunda parte desta tese, eu uso ferramentas do modelo de CQBM para analisar circuitos quânticos interagindo com curvas temporais fechadas (CTFs). Essa análise é feita traduzindo circuitos interagindo com CTFs para o modelo de CQBM e em seguida mostrando que, em alguns casos, esses circuitos podem ser transcritos como circuitos sem CTFs que realizam a mesma computação. As predições obtidas nesses casos são exatamente as mesmas daquelas obtidas usando o modelo para estudar CTFs proposto por Bennett, Schumacher e Svetlichny (BSS). Isso nos permitiu mostrar que o modelo BSS para CTFs faz predições que não concordam com aquelas dadas pelo influente modelo de CFTs proposto por David Deutsch.

List of Figures

1 The application of the unitary evolutions $Bell_{\uparrow}$ (a) or $Bell_{\downarrow}$ (b) followed by the measurement of both qubits in the computational basis is equivalent to the measurement of those qubits in the Bell basis. The circuits in (a) and (b) will be used later as part of the teleportation protocol. . p. 12

2 The teleportation protocol (Section 2.1.2). . . . . p. 13

3 The double teleporter: two teleportation protocol circuits (Fig. 2) combined to teleport two spatially separated states $|a\rangle$ and $|b\rangle$ to the same place. . . . . p. 16

4 A double teleporter with a $CNOT$ gate being applied to the teleported states. . . . . p. 17

5 Adapted version of the double teleporter: using the four-qubit state $|\chi\rangle_{abcd}$, this circuit applies the $CNOT$ gate to the unknown states $|a\rangle$ and $|b\rangle$ using only measurements and classically controlled operations. . p. 18

6 Universal graph states: (a) Square Lattice, (b) Triangular Lattice, (c) Hexagonal Lattice, (d) Kagome Lattice. . . . . p. 22

7 Graphs implementing (a) Arbitrary single-qubit unitary, (b) $CNOT$ gate, (c) CZ gate, (d) two arbitrary single-qubit rotations followed by a CZ gate and then other two single-qubit rotations. . . . . p. 30

8 (a) composed J gate teleportation protocol and (b) 1WQC protocol using adaptive measurements to implement the same unitary as in (a). . . . . p. 32

9 The simulation of a quantum circuit in a cluster state. The black dots represent qubits measured in the Z basis (and hence deleted from the cluster) and the circles with arrows represent qubits measured in the $\mathcal{B}(\theta)$ basis. Different arrow directions indicate different angles of measurement. Each set of qubits boxed with a dashed line represents the simulation of a given circuit gate. . . . . p. 35

10 Examples of graphs satisfying different flow conditions. In Fig. (a) a graph with regflow is depicted, in (b) a graph with gflow but no regflow and in (c) a graph with three different gflows: a regflow, a generalized flow and an optimal gflow (see Section 3.3.6 for more information about this example). Each vertex represents a qubit, where the black vertices represent measured qubits and the white ones are unmeasured qubits. The input qubits are represented by boxed vertices and the arrows denote the dependency structure, pointing from vertex i to its correcting set. The dashed line with an arrow represent the case where the vertex from the correcting set is a non-neighbouring vertex (vertices not linked by an edge). . . . . p. 57

11 Generic structure of extended circuits obtained from graphs with flow or gflow. See main text for information about the division into time slices. p. 64

12 Extended circuit for a simple one-way quantum computation protocol. This *J-gate identity* will be used repeatedly to simplify generic extended circuits in Chapters 5 and 6. Note that the J gate angles in (a) and (b) differ by a minus sign. . . . . p. 65

13 Two isomorphic graphs: (a) generically arranged graph and (b) same graph re-arranged to obey, from left to right, the partial ordering induced by regular flow. . . . . p. 67

14 Quantum circuit obtained from the graph in Fig. 13-b using the Star Pattern translation. . . . . p. 67

15 Fig. (a) shows a subgraph that works as a building block for the Star Pattern translation. Decomposing a general graph into these subgraphs, we can translate each one to the corresponding circuit (Fig. b) and compose them to obtain the complete translated quantum circuit. . . . p. 68

16 Using the method in Def. 16 to generate an open graph state able to simulate the circuit on the top of the figure. . . . . p. 70

17 Graphical representation of the star pattern translation being applied to the gflow pattern associated to the graph in (a). Figures (b) and (c) show the star pattern decomposition for the graph in (a). These subgraphs can be directly translated to the circuit model using the correspondence shown in Fig. 15. As a result we have the circuits in (d) and (e). Finally, composing these subcircuits we obtain the quantum circuit in (f). We can rewrite this circuit, preserving its meaning, to the circuit in (g) by adding a $SWAP$ gate acting on the (newly added) time-traveling qubit [the round wire at the bottom of Fig. (g)] and the left-most wire segment where the anachornical CZ in Fig. (f) were acting upon. By doing so, the quantum state just before the $SWAP$ gate is able to go to the future, be acted upon by the CZ gate and then go back to the past and re-enter the time respecting part of the circuit using the same $SWAP$ gate. The circuit in (g) has no anachronical CZ like the circuit in (f) but instead there is a quantum wire interacting with a closed wire (which represents a qubit traveling in time). . . . . p. 72

18 The *J-gate identity*. This is the same identity as the one shown in Sec. 4.1, and it is reproduced here for convenience of the reader. . . . . p. 77

19 Extended circuit representation of a measurement and the corrections it requires on other qubits for deterministic computation in the 1WQC model. Note that each correction arises from a particular stabilizer in the correcting set of i, raised to the power s_i, where s_i is the outcome of measurement i. This sub-circuit shows only the gates corresponding to the correcting set of qubit i. . . . . p. 79

20 Five circuit identities. The circuit identity in Figure (b) is true only if qubit k is initially in the $|+\rangle$ state. The circuit identity in Figure (d) [Figure (e)] is obtained by multiplying CZ_{ik} (CX_{ik}) in both sides of the identity in Figure (a) [Figure (c)]. . . . . p. 80

21 Rewrite Procedure 1. In Fig. 21-a we show the undesired CZ_{ik} in time slice $\mathcal{C}_i$ with the corresponding, initial entangling-round CZ_{jk} in time slice $\mathcal{E}_1$. In Fig. 21-b the CZ_{jk} gate was moved from $\mathcal{E}_1$ to $\mathcal{C}_i$ where the circuit identity in Fig. 20-a can be applied, resulting in the circuit depicted in Fig. 21-c. . . . . p. 82

22 Stepwise influencing path $\wp_i(j)$. See Def. 21. . . . . p. 96

23 Extending a stepwise influencing path ending at vertex v according to Lemma 7. . . . . p. 98

24 Rewrite Procedure 2. This RP is composed by several applications of the rewrite rule in Figure 20-d. Although all E gates are drawn in the $\mathcal{E}$ slice, the requirement for this RP to be applied is that those E gates are placed just before the corresponding CX gates (that is, with no gate in between). . . . . p. 99

25 Rewrite Procedure 3. If $L_s(k) = L_s(i)$, slices $(\mathcal{E}, \mathcal{J}, \mathcal{C})$ and $(\mathcal{E}', \mathcal{J}', \mathcal{C}')$ become the same; The rewrite procedure remains exactly the same. . . p. 99

26 Rewrite Procedure 4. If $L_s(k) = L_s(i)$, slices $(\mathcal{E}, \mathcal{J}, \mathcal{C})$ and $(\mathcal{E}', \mathcal{J}', \mathcal{C}')$ become the same; The rewrite procedure remains exactly the same. . . p. 100

27 Removing even many CZ_{ij} from a SSF extended circuit. See Lemma 8 for more information. . . . . p. 102

28 Using compactification procedures to obtain J-gate parallelization for circuits with Pauli angles. See example 2 for more information. . . . . p. 113

29 Simplifying an extended circuit originated from the entanglement graph with gflow of Fig. (a). For a step-by-step explanation, see section 5.4.2. p. 115

30 Simplifying the extended circuit originated from the graph in Fig. (a). For a step-by-step explanation, see Sect. 5.4.3. . . . . p. 116

31 Aplication of the procedure described in Def. 16 to the circuit in Figure 32-a. See Step 1 of Sec. 6.1.1 for more information. . . . . p. 121

32 A complete example of global circuit optimization. In each figure the shaded gates are the new gates obtained through the RPs (see Section 6.1.1 for more information). The angles of each J-gate are arbitrary and they have been omitted from the figure. . . . . p. 125

33 A summary of the optimization procedure using extended translation and compactification. Each arrow on the left of the table indicates a different optimization or translation procedure. The parallelization procedure described in [23] (without Pauli simplification) ends with the "Parallelized Circuit" whereas our optimization can end with a "Compactified Circuit" or a "Parallelized Compactified Circuit" depending the goal of the optimization. . . . . p. 127

34 a) CTC takes qubit back in time to interact with its past self; b) Bennett/Schumacher/Svetlichny (BSS) circuit to simulate this CTC probabilistically, using teleportation and post-selection (see main text). . . . p. 134

35 Rewritten BSS circuit using preparation of and projections onto $|+\rangle$ states. The unitary V is decomposed using the universal gate-set consisting of $J(\theta)$ and CZ. . . . . p. 135

36 BSS circuit simulating a CTC with unitary $V = [J(-\theta) \otimes I]CZ$. . . . . p. 135

37 Two equivalent circuits, i.e. implementing the same unitary. In a) we have a classically-controlled X unitary dependent on the measurement outcome of $|\pm_\theta\rangle$ basis projection. In b) this has been turned into a coherent circuit with measurement onto the Z basis. . . . . p. 137

38 a) Circuit that includes a CTC with an anachronical CZ gate; it is equivalent to the two circuits in Fig. 37. b) The same circuit rewritten in the BSS format. . . . . p. 137

39 Circuit that implements the probabilistic BSS simulation of the CTC circuit in Fig. 38-b. . . . . p. 138

40 (a) Entanglement graph corresponding to state $|G\rangle$ in Eq. (7.4). On this state we can perform the sequence of one-way operations in Eq. (7.3). The extended translation of Eqs. (7.8), (7.9) and (7.10) are shown in (c), (e) and (g), respectively. In Figs. (d), (f) and (h) I redraw the CTC circuits in Figs. (c), (e) and (g), respectively, so as to explicitly show the CTC as a time-travelling qubit. . . . . p. 140

41 Translating the BSS circuit in Fig. 39 to the one-way model (see Sec. 7.4.1). . . . . p. 144

42 Stabilizer manipulations that simplify the one-way sequence (7.11); $\langle_i+|$ denotes a projection onto $|0\rangle + i|1\rangle$. a) Graph representing initial entanglement structure and operations. Qubit 3 is the input and qubits 1 and 5 are measured in the X basis. b) Effect of local complementation (LC) on qubit 5; c) LC on qubit 1; d) Pauli Z deletion of qubit 1; 3) LC on qubit 5; f) Pauli Z deletion of qubit 5. The final pattern represents the one-way two-qubit implementation of the $J(-\theta)$ gate, see Fig. 37. . . . p. 145

43 An analysis of the stabilizers in eqs. (7.16) and (7.17) indicates that the graph on the right represents state $|\psi\rangle$ and the one on the left state $|\phi\rangle$ (see main text). The local complementation unitary $C_1 S_2^\dagger S_3^\dagger$ changes the graph on the left into the graph on the right. Local complementations change the state without changing its entanglement structure or the one-way computations implementable by it. . . . . p. 147

44 a) Deutsch's model for a CTC. b) This is the relationship between Deutsch's unitary U and unitary V in the BSS CTC circuit of Fig. 34-b. . . . . p. 148

45 Three representations for the same CTC circuit. a) Deutsch formulation. b) BSS formulation. c) Short-hand form of either. . . . . p. 149

46 Deutsch's formulation of the extended CTC circuit of Fig. 38-b. . . . . p. 150

Contents

1 Introduction p. 1

1.1 General picture and structure of this thesis . . . . . p. 4

1.2 List of publications and corresponding chapters . . . . . p. 8

1.3 Notation . . . . . p. 9

2 Quantum computation driven by measurements p. 10

2.1 Teleportation-based quantum computation . . . . . p. 11

2.1.1 Bell states . . . . . p. 11

2.1.2 The teleportation protocol . . . . . p. 12

2.1.3 Teleportation as a quantum computing primitive . . . . . p. 14

2.1.3.1 Teleporting a single-qubit gate . . . . . p. 14

2.1.3.2 The CNOT gate . . . . . p. 15

2.2 The rise of measurement-based models . . . . . p. 18

2.3 A brief description of the one-way QC model . . . . . p. 19

2.4 Resource states . . . . . p. 22

2.4.1 Cluster states and graph states . . . . . p. 23

2.5 Measuring graph state qubits . . . . . p. 26

2.5.1 Single-qubit measurements . . . . . p. 28

2.5.2 Removing redundant qubits . . . . . p. 29

2.5.3 Moving the logical state of a qubit in a graph . . . . . p. 29

2.6 Simulating the circuit model . . . . . p. 30

2.6.1 Universal single-qubit rotations and the need for adaptive measurements . . . p. 31
2.6.2 Two-qubit gates . . . p. 33
2.6.3 Simulating arbitrary quantum circuits . . . p. 34
2.6.4 Beyond quantum circuit simulations . . . p. 36
2.6.5 Simulating quantum circuits with unbounded fan-out . . . p. 36
2.7 Clifford computations . . . p. 37
2.7.1 Pauli measurements and the Clifford group . . . p. 37
2.7.2 The Gottesman-Knill theorem . . . p. 38
2.7.3 Quantum-classical tradeoff . . . p. 39

3 Deterministic computation using quantum measurements p. 41
3.1 The measurement calculus . . . p. 43
3.1.1 Commands . . . p. 43
3.1.2 Measurement patterns . . . p. 44
3.1.3 Command identities . . . p. 47
3.1.4 Standardization . . . p. 48
Example 1: Teleportation . . . p. 49
Example 2: Single-qubit rotation . . . p. 50
3.2 Determinism in 1WQC . . . p. 50
3.2.1 Types of determinism . . . p. 54
3.3 Flow theorems . . . p. 55
3.3.1 Motivation . . . p. 56
3.3.2 Regular flow . . . p. 56
3.3.3 Generalized flow . . . p. 58
3.3.4 Maximally-delayed Generalized flow . . . p. 59
3.3.5 Constructing a measurement pattern from a flow function . . . p. 60

3.3.6 Examples . . . . . p. 60

4 Translating one-way patterns into the circuit model p. 62

4.1 Extended translation . . . . . p. 63

4.2 Star pattern translation . . . . . p. 66

4.2.1 From measurement patterns to circuits . . . . . p. 66

4.2.2 From circuits to measurement patterns . . . . . p. 69

4.3 The problem of using star pattern translation for patterns with no regular flow . . . . . p. 71

4.4 Alternative approaches to the translation problem . . . . . p. 74

5 Compact circuits from one-way quantum computation p. 75

5.1 Compactification procedures and circuit rewriting rules . . . . . p. 76

5.1.1 Compactification procedures . . . . . p. 77

5.1.2 Circuit rewrite rules . . . . . p. 78

5.2 Compact circuits from graphs with regular flow . . . . . p. 80

5.2.1 Regular flow structural properties . . . . . p. 81

5.2.2 A compactification algorithm for regular flow extended circuits . p. 81

5.2.2.1 A rewrite procedure for regular flow . . . . . p. 82

5.2.2.2 The algorithm . . . . . p. 83

5.2.2.3 Comparison with star pattern translation . . . . . p. 85

5.3 Compact circuits from graphs with signal-shifted Flow . . . . . p. 85

5.3.1 Signal-shifting . . . . . p. 86

5.3.1.1 Rewrite rules . . . . . p. 86

Example . . . . . p. 87

5.3.1.2 Algorithm . . . . . p. 87

5.3.2 The signal-shifted flow . . . . . p. 90

5.3.3 SSF structural properties . . . . . p. 95

5.3.4 Rewrite procedures for signal-shifted flow . . . . . p. 98
5.3.5 Compactification procedure for signal-shifted flow . . . . . p. 106
5.3.6 Discussion of results and possible extensions . . . . . p. 111
Example: Pauli measurements . . . . . p. 111
5.4 Compact circuits from graphs with generalized flow . . . . . p. 113
5.4.1 Introduction . . . . . p. 114
5.4.2 First example . . . . . p. 115
5.4.3 Second example . . . . . p. 116
5.4.4 Discussion . . . . . p. 116
5.5 Concluding remarks . . . . . p. 117

6 Optimizing quantum circuits using one-way quantum computation techniques p. 118

6.1 Optimizing quantum circuits by back-and-forth translation . . . . . p. 118
6.1.1 Example . . . . . p. 120
Step 1 . . . . . p. 122
Step 2 . . . . . p. 122
Step 3 . . . . . p. 123
Step 4 . . . . . p. 124
6.2 Complexity analysis and discussion . . . . . p. 126

7 Closed timelike curves in one-way quantum computation p. 130

7.1 Review of the literature . . . . . p. 131
7.2 A model for CTCs based on teleportation and post-selection: The BSS model . . . . . p. 133
7.3 Closed timelike curves in one-way quantum computation . . . . . p. 136
7.3.1 Obtaining CTC-assisted circuits from one-way patterns . . . . . p. 139
7.4 Deterministic simulations of CTCs . . . . . p. 141

7.4.1 Example . . . . . p. 144
7.5 Comparing the BSS model with Deutsch's model . . . . . p. 148
7.5.1 The Deutsch model . . . . . p. 148
7.5.2 Comparison with the BSS model . . . . . p. 149
7.5.3 Discussion . . . . . p. 151
7.6 One-way model: a toy model for space-time? . . . . . p. 152

8 Conclusion and further research directions p. 155

p. 159

1 Introduction

Quantum computers are expected to outperform their classical counterparts. Since Richard Feynman's seminal 1982 paper [46], where the idea of a computer based on quantum physics was first suggested, several different models for quantum information processing have been proposed. Since there is no rigorous proof that quantum computation is more powerful than classical computation, a lot of work has been done trying to understand which physical properties give quantum computers their apparent power.

A number of useful quantum algorithms are known, including algorithms for factoring [105, 110], searching unstructured databases [57] and to simulate quantum systems [13, 74, 47, 16, 72, 69]. However, the lack of understanding about what physical properties enable a quantum computer to run algorithms that outperform classical ones makes further development of quantum algorithms a great challenge. A good approach to improve this understanding is the development and analysis of different models for quantum computing, in special those relying on different physical properties to drive the computation.

There are several models for quantum computing, such as the quantum circuit model [38, 86], adiabatic quantum computation [45, 68, 17, 101], and measurement-based quantum computation (MBQC) models such as teleportation-based quantum computation [52], ancilla-driven quantum computation [104, 11, 67] and the one-way quantum computation (1WQC) model [95, 21, 97, 100, 20]. Those models were proved to be polynomialy equivalent in terms of depth and spatial resource, but the key elements and requirements for their physical implementation vary significantly. A particularly interesting class of models is the one that rely on measurements to drive the computation, in special the 1WQC model. In this model entanglement is a resource provided at the beginning and which is destroyed by single-qubit measurements, whose correlations enable the computation.

In this thesis I study the one-way model, in particular several issues that arise when we translate computations from the circuit model to 1WQC model and vice-versa, with the purpose of getting new insights on how the algorithms work and what key elements are

required for the implementation of a given algorithm. I develop a new translation framework able to translate to the circuit model a large class of 1WQC algorithms. The main difference with respect to other translation procedures is that the method described in this thesis preserves some features of the 1WQC algorithm as, for example, the dependency structure between the gates/operations. It is clear that having translation procedures able to preserve different properties can just enrich the understanding on which resources are needed to perform a given computation.

I also give two applications that profit from the methods developed for optimizing and translating one-way computations to the circuit model. The first is a circuit optimization scheme based on back-and-forth translation between the 1WQC model and the circuit model. Quantum circuit optimization is a subject of great importance for quantum computing. The enormous technological effort required to perform (even simple) quantum computing tasks makes the research on circuit optimization a potential shortcut to more complex quantum computing tasks, bringing quantum algorithms down to the current technological capabilities.

I contribute to this line of research by introducing an automated circuit optimization procedure. This procedure is based on the translation of a given quantum circuit into a one-way quantum computation. These two models utilize remarkably different information processing tools: while the former is based on unitary evolution of an initially non-entangled set of qubits, the latter needs an initial highly-entangled multi-qubit state, where the information processing is driven by measurements only. Naturally, since the two aforementioned models use different information processing tools, each model has its own optimization techniques.

In the 1WQC model, for instance, most of the optimization techniques are based on the identification of a more efficient correction structure that is directly linked to the geometry of the underlying global entanglement structure. Examples of these techniques are *signal shifting* [35] and *generalised flow* [24]. The so-called *standardisation* procedure [35] can also reduce the number of computational steps by rearranging the 1WQC operations into a normal form. Moreover, all Pauli measurements in this model can be performed at the beginning of the computation [96], which is a surprising difference from the quantum circuit model.

On the other hand, most optimization techniques for quantum circuits are based on template identification and substitution. For instance in [49], some circuit identities are used to modify the teleportation and dense coding protocols, with the purpose of giving a

more intuitive understanding of those protocols. Similarly in [103] and [79] a set of circuit identities for reducing the number of gates in the circuit for size optimization was given. In contrast to that, in [84] a useful set of techniques for circuit parallelization was provided, where the number of computational steps is reduced by using additional resources. However, as noted in [103], all the aforementioned circuit optimization techniques are basically exchanging a sequence of gates for a different one without any consideration of the structure of the complete circuit being optimised. The translation into 1WQC would allow us to explore the global structure of a given circuit.

The first such optimization scheme by back and forth translation between the two models was presented in [23]. However the backward translation into the circuit required the addition of many ancilla qubits. In this thesis I propose a new automated optimization scheme able to translate the optimized computation back to the circuit model without adding any ancillas. We start by translating the circuit to the 1WQC model using a technique that gives a graph whose geometry reflects the gate arrangement in the associated circuit. We then use a 1WQC optimization technique (known as signal shifting) able to reduce the depth of the computation[1]. It is important to note that, since this optimization technique relies on global properties of the graph (related to its geometry), the optimization implemented in the circuit model is related to the global arrangement of gates in the original circuit. The optimized 1WQC computation is then translated to the circuit model using a new translation procedure (developed in Chapter 5), resulting in a circuit with as many wires as the original one but with smaller number of computational time slices.

The second application is related to a more fundamental topic, namely the existence and computational power of closed timelike curves. The possibility of time travel has been studied for decades in the context of general relativity. Several decades after Einstein developed the general theory of relativity, the first explicit spacetime geometry containing closed time-like curves (CTCs) - paths in spacetime that would allow a physical system to interact with its former self - was proposed by Kurt Gödel [50]. After Gödel's work, a variety of spacetimes containing CTCs were proposed [18, 51]; in fact, it was later realized that CTCs are a generic feature of highly curved, rotating spacetimes [18, 109, 71].

Assuming that closed timelike curves exist, a series of results were obtained regarding their implications for quantum mechanics and quantum computation [39, 12, 28, 15, 92, 93]. In this thesis I describe how the one-way model of measurement-based quantum

[1]That is, the same computation can be implemented faster, by increasing the number of operations that can be performed simultaneously.

computation [95] encompasses in a natural way a model for CTCs proposed by Bennett and Schumacher [1], and more recently (and independently) by Svetlichny [106]. I show that the one-way model effectively simulates deterministically a class of CTCs in this model, and characterize this class. A second model for CTCs is Deutsch's highly influential study of quantum time-travel [39]. I show that Deutsch's model leads to predictions conflicting with those of the one-way model, and identify the reason behind this.

1.1 General picture and structure of this thesis

Here I summarize the main contributions of this thesis and discuss how this thesis is organized. In the first chapters I review the key ideas of the one-way quantum computation model, with a special attention to the elements necessary for understanding the original work presented in later chapters.

- **Chapter 2.** I start by discussing the seminal work by Gottesman and Chuang [52] where they show that an adaptation of the teleportation protocol [14] can be used as a quantum computing primitive. Their scheme, known as *gate teleportation* or *teleportation-based quantum computation*, contains the main ingredients of a measurement-based quantum computation model: preparation of a (sufficiently large) highly entangled state and measurements on bases which may depend on previously obtained outcomes (adaptive measurements). I proceed with a brief overview of other measurement-based models that were proposed after [52] and then I present the one-way quantum computation model [95, 21, 97]. In Sec. 2.4 I give some examples of quantum states that can be used as resource states in a one-way computer, giving special attention to the widely used *graph states*. In Sec. 2.5 I show the effect of performing single-qubit measurements in graph states and in Sec. 2.6 I show how quantum circuits can be simulated in the one-way model. Finally, in Sec. 2.7 I discuss the role of Pauli measurements in the one-way model; these measurements have the interesting property of being independent of any other measurement in a one-way quantum computation. As a consequence, all Pauli measurements can be performed at once in the first computational time slice.

One of the most important concepts behind the one way model is the way intrinsically random quantum measurements can be harnessed as a tool to implement deterministic computation. Explaining this and some tools to design 1WQC algorithms is the main goal of the next chapter.

- **Chapter 3.** In Sec. 3.1 I review the so-called *measurement calculus* [35], which is a formal language developed to represent and analyze 1WQC algorithms. In Sec. 3.2 I describe the different types of determinism that a 1WQC algorithm can be required to have, explaining why I am going to require just one of those types of determinism. Next, in Sec. 3.3, I review several correcting strategies generically known as *flows*. A flow is a set of conditions over a graph (representing the initial entanglement structure) that verify whether there exists a partial ordering to perform the measurements so that a deterministic computation can be implemented. I review three different types of flow: (1) *Regular Flow* [33], (2) *Generalized Flow* [24] and (3) *Maximally-delayed Generalized Flow* [81]. Those properties are not mutually exclusive and therefore a graph can have two or more types of flow. A comparison between all the aforementioned flows (together with some examples) concludes the chapter.

The one-way quantum computation model is equivalent to the circuit model. This means that every problem that is solvable in one model, can also be solved by the other with at most a polynomial overhead in the number of computational steps. However, a computation in the circuit model requires completely different resources than the corresponding computation in the one-way model. Based on this notion, most of the original work presented in this thesis aims at (i) developing new translation procedures from the one-way model to the circuit model (Chapter 5) and (ii) translating computations from one model to another in order to get insights that would be hard to get when analyzing just one model (Chapters 6 and 7).

- **Chapter 4.** In this chapter I review two different translation procedures and explore their limitations and advantages. In Sec. 4.1 I review and extend the definition of *extended circuits*, which are easily obtainable translations of measurement patterns. As we shall see, this is a straightforward method but it is inefficient in some important ways. Next, in Sec. 4.2, I describe the so-called *star pattern translation* (or simply SPT). The SPT is a translation framework that gives circuits which are less demanding (in terms of number of qubits) than the corresponding extended circuits. The SPT is, however, a very limited translation framework since it works only for measurement patterns obtained from a regular flow (Sec. 3.3.2). In Sec. 4.2.2, I show how the SPT can be used "backwards" to translate circuits to the 1WQC model, resulting in measurement patterns with regular flow. In Sec. 4.3, I explain exactly why and in which cases the SPT fails. The insights contained in

that section were the starting point for the development of a new translation framework (introduced in Chapter 5 and with an application given in Chapter 6) and the study of the simulation of closed timelike curves using the 1WQC model (Chapter 7).

- **Chapter 5.** This chapter is divided as follows. In Section 5.1 I introduce the concept of compact circuits and the method for obtaining them from extended circuits, which I call *compactification procedures*. Section 5.2 is reserved for the development of the compactification procedure for regular flow extended circuits. First, in Sec. 5.2.1, I review the regular flow definition and comment on some important properties of regular flow extended circuits. The algorithm that implements the compactification procedure for regular flow is introduced in Sec. 5.2.2; A comparison with the existing method for regular flow translation, known as the Star Pattern Translation (see Sec. 4.2.1), and an example of the algorithm running are also provided in this section. In the beginning of Section 5.3, I introduce a new type of flow that I call Signal-shifted Flow (SSF) and explore several of its structural properties. The SSF is more general than regular flow and its limitations and advantages are explored in the section. An algorithm to implement the compactification procedure for SSF extended circuits is given in Sec. 5.3.5. Finally, in Sec. 5.4 I comment on the difficulty of designing compactification procedures for arbitrary flows and give a couple of examples of generalized flow extended circuits being transformed into their compact versions.

In the next two chapters I present two problems whose understanding profits from the translation between models that I develop. In Chapter 6 I present a quantum circuit optimization procedure. The procedure consists in three parts: (i) translation of a circuit to the 1WQC model, (ii) application of optimization techniques to the 1WQC algorithm obtained in (i), and (iii) translation of the optimized 1WQC algorithm back to the circuit model. The second successful case where the approach of translating algorithms from one model to another brought some new insights, is related to the so-called closed timelike curves. The analysis of quantum circuits interacting with closed timelike curves is the subject of Chapter 7.

- **Chapter 6.** In this chapter I give an application of the SSF compactification procedure introduced and explored in Chapter 5, namely the optimization of quantum circuits. In Sec. 6.1, I describe how a compactification procedure can be used -

together with other techniques - to optimize quantum circuits and give a full example of such procedure. I conclude this chapter with a complexity analysis (Sec. 6.2) of the optimization technique proposed here, comparing it with other known techniques.

- **Chapter 7.** I start by giving a brief summary of the recent revival of interest in closed timelike curves (CTCs) in the quantum information community, highlighting the main contributions over the last years. Then, in Sec. 7.2, I review the CTC model based on quantum teleportation and postselection proposed by Bennett, Schumacher and Svetlichny [1, 106]. In Sec. 7.3 I discuss how CTCs appear naturally in the one-way model, and show that they correspond to CTCs in the Bennett/Schumacher/Svetlichny model. In Sec. 7.4 I characterize a class of CTC-assisted quantum circuits that can be rewritten as time-respecting circuits (*i.e.*, with no CTCs). I give an explicit method for verifying if a CTC-assisted circuit can be simulated using the 1WQC model. Next, in Sec. 7.5, I review the highly influential CTC model proposed in 1991 by David Deutsch [39]; I use Deutsch's consistency conditions to calculate the predictions for the same CTC-assisted circuits analyzed in Sec. 7.2 using the BSS model. I show that while the BSS's model encompasses the one-way model prediction, Deutsch's model gives completely different predictions. Finally, in Sec. 7.6 I discuss a recent proposal by Raussendorf *et al.* [99] for using 1WQC as a quantum mechanical toy model for space-time.

- **Chapter 8.** In this final chapter, I give my concluding remarks about the work presented in this thesis and point out some further research directions and open problems.

To conclude this section, I would like to make a disclosure. Some of the results presented in Sec. 5.3, which is based in [42], are due to one of my collaborators in that work, Einar Pius. More specifically, the algorithm in Sec. 5.3.1.2 and most of the proofs in Secs. 5.3.2 and 5.3.3 were due to him and presented in our joint work [42]. Since those results are indispensable for understanding the rest of the results in Sec. 5.3, I have opted to reproduce (with a few changes and adaptations) part of the text from our joint work [42].

1.2 List of publications and corresponding chapters

I list below the articles (published or submitted) written to communicate the original work that I present in this thesis. At the end of each item I inform in which chapter(s) of this thesis one can find the content related to the corresponding article.

1. Raphael Dias da Silva, Ernesto F. Galvão and Elham Kashefi. **Closed time-like curves in measurement-based quantum computation** *Phys. Rev. A 83, 012316* (2011); [Content related to this publication can be found in Chapter 7].

2. Raphael Dias da Silva and Ernesto F. Galvão. **Compact quantum circuits from one-way quantum computation**. *Phys. Rev. A 88, 012319* (2013); [Content related to this publication can be found in Chapters 4 and 5].

3. Raphael Dias da Silva, Einar Pius and Elham Kashefi. **Global quantum circuit optimization**. Submitted. *arXiv:1301.0351 [quant-ph]* (2013); [Content related to this preprint can be found in Chapters 5 and 6].

1.3 Notation

In this section I summarize the main notation and symbols that will be used throughout this thesis. I list below the single-qubit gates that I will refer to in this thesis by its name or symbol (in rare cases I will explicitly show its matrix form). In order of appereance, the gates are: Hadamard (H), phase (S), $\frac{\pi}{8}$-*gate* (T) and arbitrary phase gate [$P(\theta)$]:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \quad P(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}.$$

I will also constantly refer to the so-called *Pauli matrices* or, in the context of quantum computing, *Pauli gates*:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The following two-qubit gates will also be constantly used throughout this thesis: Controlled-Z (CZ), Controlled-NOT (CX) and the $SWAP$ gate:

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

2 Quantum computation driven by measurements

Quantum computation was first conceived as a generalization of classical computing: most quantum models were built upon key concepts from classical information processing architectures, as it is the case for quantum Turing machines, quantum circuits and quantum random walks. These models use reversible, unitary evolution to drive the information processing, whereas measurements take place at the end to read out classical information. Due to the irreversible and probabilistic character of quantum measurements, it is quite surprising that it can be used as main tool in a quantum computing model. However, in 1999, Gottesman and Chuang [52] described how one could apply arbitrary quantum gates via measurements, using an adaptation of the quantum teleportation protocol [14]. This approach was further developed by other researchers [87, 73, 63, 64], enabling one in principle to perform arbitrary computations given a few primitives: preparation of maximally entangled systems of fixed, small dimension; multi-qubit measurements on arbitrary sets of qubits; and the possibility of adapting the measurement bases depending on earlier measurement outcomes. These models of computation draw on measurements to implement the dynamics, and as such are collectively called measurement-based quantum computation (MBQC).

One of the most prominent MBQC models, the so-called one-way quantum computation (1WQC) model, was developed by Robert Raussendorf and Hans Briegel in 2001 [95]. The one-way model has some remarkably interesting properties: in this model, any algorithm can be implemented using a generic resource state called the *cluster state*, upon which adaptive single-qubit measurements are performed. Moreover, since no entangling measurement is performed in this model (as is necessary in teleportation-based models), the contribution of entanglement for the quantum speedup, one of the main fundamental questions in quantum computing, can be more systematically analyzed by studying which quantum states are useful resource states.

In this chapter I review the main concepts of measurement-based QC models, giving special attention to the 1WQC model, which is the backbone of this thesis. First, in Section 2.1, I review the well-known teleportation protocol and explain how it can be adapted to be used as a quantum computing primitive. This modified teleportation protocol was the first proposed MBQC model and, as such, it contains the main ideas behind MBQC models in general. In Section 2.2, I give a brief summary of the different MBQC models that appeared in the literature soon after the teleportation-based model. In Section 2.3, a brief overview of the one-way model is given, whereas an in-depth treatment of the concepts discussed in this section will be given in the following sections. In Section 2.4, I discuss which resource states are useful for the 1WQC model, showing examples of universal resource states and how to create the widely used graph states. In Section 2.5, I review projective measurements and show the effect of measuring graph state qubits in different bases. In Section 2.6, I explain how information is processed in the 1WQC model and how it can simulate the circuit model. Finally, in Section 2.7, I discuss the role of the so-called Clifford gates, which surprisingly can be implemented all at once in the beginning of the computation in the 1WQC model.

2.1 Teleportation-based quantum computation

Many of the important properties of MBQC models are present in the teleportation protocol [14] itself: the preparation of an entangled state to be used as a resource; the necessity of communicating and/or storing classical data (measurement outcomes) and also the local corrections needed to account for the intrinsic random character of measurement. Let us briefly review how the teleportation protocol works and then explain how it can be used as a primitive for quantum computing.

2.1.1 Bell states

Bell states are maximally-entangled two-qubit states widely used in several quantum information protocols. Those states were named after John Bell, who explored via the so-called *Bell inequalities* many interesting properties of such states. They are also known as *EPR-like states*, after Einstein, Podolsky and Rosen, which used similar states in a thought experiment designed to criticize some aspects of quantum theory [44].

Bell states can be constructed in the following way: (1) Prepare the four computational basis states $|00\rangle, |01\rangle, |10\rangle$ and $|11\rangle$; (2) apply to each of those states the operator $U =$

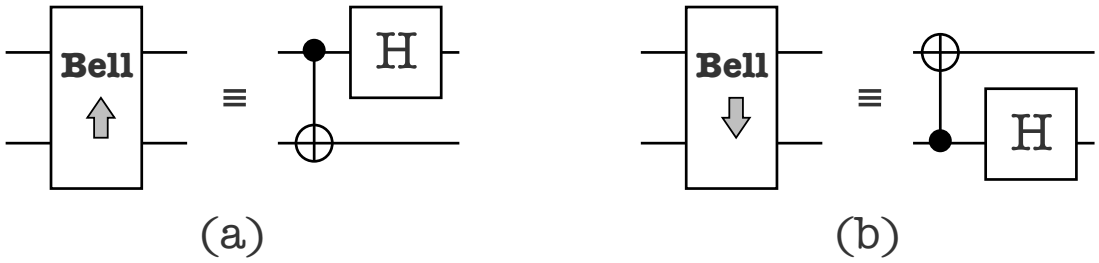


Figure 1: The application of the unitary evolutions $Bell_{\uparrow}$ (a) or $Bell_{\downarrow}$ (b) followed by the measurement of both qubits in the computational basis is equivalent to the measurement of those qubits in the Bell basis. The circuits in (a) and (b) will be used later as part of the teleportation protocol.

$(H \otimes I)CNOT$, where H is the Hadamard gate and I is the Identity operator. In Fig. 1 I give two circuits able to create Bell states, using a graphical notation that will be used later in this Chapter. The resulting states are the following:

$$|\beta_{00}\rangle \equiv \frac{1}{\sqrt{2}}\big(|00\rangle + |11\rangle\big) \tag{2.1}$$

$$|\beta_{01}\rangle \equiv \frac{1}{\sqrt{2}}\big(|00\rangle - |11\rangle\big) \tag{2.2}$$

$$|\beta_{10}\rangle \equiv \frac{1}{\sqrt{2}}\big(|01\rangle + |10\rangle\big) \tag{2.3}$$

$$|\beta_{11}\rangle \equiv \frac{1}{\sqrt{2}}\big(|01\rangle - |10\rangle\big) \tag{2.4}$$

which can be succinctly represented as $|\beta_{vw}\rangle \equiv \frac{1}{\sqrt{2}}\big(|0v\rangle + (-1)^w|1\bar{v}\rangle\big)$, where $v, w \in \{0, 1\}$ and $\bar{v} = 1 - v$. Since U is a unitary operator, the four Bell states form a basis for the Hilbert space of a two-qubit system. A measurement onto this basis is called a *Bell measurement*. Note also that any Bell state can be obtained from any other Bell state by applying Pauli operators:

$$|\beta_{vw}\rangle = Z^w X^v \otimes I|\beta_{00}\rangle = I \otimes Z^w X^v|\beta_{00}\rangle \tag{2.5}$$

up to a global phase. In other words, if two parties share a Bell state, each party has the ability of transforming it into another Bell state by applying local Pauli operators.

2.1.2 The teleportation protocol

The teleportation protocol [14] consists basically of three steps: (1) An EPR pair is shared between Alice and Bob; (2) Alice, who wants to send an unknown state $|\phi\rangle$ to Bob,

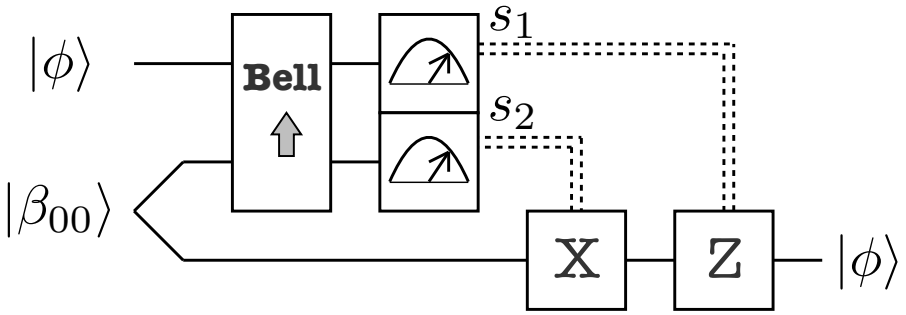


Figure 2: The teleportation protocol (Section 2.1.2).

performs a joint measurement in the Bell basis using one qubit of the shared EPR pair together with the qubit in the $|\phi\rangle$ state; (3) Finally, Alice communicates the measurement outcome to Bob (in the form of two classical bits $s_1 = w$ and $s_2 = v$) who then applies the corrections X^v and Z^w to his qubit of the EPR pair. A representation of the protocol is depicted in Fig. 2.

Let $|\psi_0\rangle$ be the initial state of the teleportation circuit shown in Fig. 2. It is a three-qubit state composed by a Bell state, $|\beta_{00}\rangle$, and the state $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are unknown amplitudes:

$$|\psi_0\rangle \equiv \frac{1}{\sqrt{2}}\Big[\alpha|0\rangle_1\big(|00\rangle + |11\rangle\big)_{23} + \beta|1\rangle_1\big(|00\rangle + |11\rangle\big)_{23}\Big] \tag{2.6}$$

Alice uses the qubit she wants to send to Bob as the control for a CNOT gate whose target is her EPR qubit. The resulting state is the following:

$$|\psi_1\rangle_{123} \equiv \big(CNOT_{12} \otimes I_3\big)|\psi_0\rangle_{123} = \frac{1}{\sqrt{2}}\Big[\alpha|0\rangle_1\big(|00\rangle + |11\rangle\big)_{23} + \beta|1\rangle_1\big(|10\rangle + |01\rangle\big)_{23}\Big] \tag{2.7}$$

Next, she applies the Hadamard gate to the first qubit:

$$\begin{aligned}
|\psi_2\rangle_{123} &\equiv \big(H_1 \otimes I_2 \otimes I_3\big)|\psi_1\rangle_{123} && (2.8)\\
&= \frac{1}{2}\Big[\alpha\big(|0\rangle + |1\rangle\big)\big(|00\rangle + |11\rangle\big) + \beta\big(|0\rangle - |1\rangle\big)\big(|10\rangle + |01\rangle\big)\Big] && (2.9)\\
&= \frac{1}{2}\Big[|00\rangle\big(\alpha|0\rangle + \beta|1\rangle\big) + |01\rangle\big(\alpha|1\rangle + \beta|0\rangle\big)\\
&+ |10\rangle\big(\alpha|0\rangle - \beta|1\rangle\big) + |11\rangle\big(\alpha|1\rangle - \beta|0\rangle\big)\Big] && (2.10)
\end{aligned}$$

Finally, Alice measures her qubits in the computational basis $\{|00\rangle, |10\rangle, |01\rangle, |11\rangle\}$, which collapses Bob's qubit into one of the following states (depending on Alice's measurement outcomes, represented by the bits w and v):

Alice measurement's outcome (wv)	Bob's resulting state
00	$\alpha\|0\rangle + \beta\|1\rangle$
01	$\alpha\|1\rangle + \beta\|0\rangle$
10	$\alpha\|0\rangle - \beta\|1\rangle$
11	$\alpha\|1\rangle - \beta\|0\rangle$

Therefore, Bob is able to recover Alice's $|\phi\rangle$ state in his lab by applying the correction operator $X^v Z^w$ to the qubit he possesses, where $v, w \in \{0, 1\}$ are the two classical bits sent by Alice.

2.1.3 Teleportation as a quantum computing primitive

In [52], Gottesman and Chuang showed that the teleportation scheme can be used as a computational primitive if the initial state used as resource state can be chosen appropriately. By doing so, it is possible to teleport a quantum state "through" a logical gate U, that is, instead of obtaining the final state $Z^w X^v|\psi\rangle$, one would obtain $Z^w X^v U|\psi\rangle$, which, after the Pauli corrections, is exactly the quantum state one would obtain by directly applying gate U to the initial state. In fact, this "trick" can be done for U of any dimensionality (as long as the initial entangled state has the appropriate dimensionality), giving as output states of the form $P^s U|\psi\rangle$, where P is a collection of Pauli operators depending on the measurements outcomes (jointly represented as s). In what follows we show how to implement gates from the universal gate-set $\{R_z(\theta), R_x(\theta), CNOT\}$[1], using the teleportation scheme as a primitive.

2.1.3.1 Teleporting a single-qubit gate

The implementation of single-qubit gates is achieved by a simple modification of the measurement basis used in the original teleportation protocol. Instead of using the Bell basis [formed by the states in Eq. (2.5)] to perform Alice's joint measurement, we use a different basis, called the *"rotated Bell basis"*:

$$|\beta(U)_{vw}\rangle = U^\dagger \otimes I|\beta_{vw}\rangle. \tag{2.11}$$

Now we re-do the calculation of the teleportation protocol circuit but using the rotated basis in Eq. (2.11) instead of the original Bell basis. Projecting the state in Eq. (2.6)

[1] where $R_z(\theta)$ and $R_x(\theta)$ are rotations by an angle θ around the X and Z axis, respectively.

onto the rotated Bell state $|\beta(U)_{00}\rangle$ yields:

$$
\begin{aligned}
{}_{12}\langle\beta(U)_{00}||\psi_{00}\rangle_{123} &= {}_{12}\langle\beta(U)_{00}||\phi\rangle_1|\beta_{00}\rangle_{23} && (2.12)\\
&= {}_{12}\langle\beta_{00}||U\phi\rangle_1|\beta_{00}\rangle_{23} && (2.13)\\
&= \frac{1}{\sqrt{2}}{}_{12}\langle\beta_{00}|\Big[\alpha U|0\rangle_1\big(|00\rangle+|11\rangle\big)_{23} && (2.14)\\
&+ \beta U|1\rangle_1\big(|00\rangle+|11\rangle\big)_{23}\Big]\\
&= \frac{1}{2}\Big(\alpha U|0\rangle+\beta U|1\rangle\Big)_3 && (2.15)\\
&= \frac{1}{2}U|\phi\rangle_3 && (2.16)
\end{aligned}
$$

where I used $\alpha U|0\rangle+\beta U|1\rangle \equiv U(\alpha|0\rangle+\beta|1\rangle)$. Note that the state in Eq. (2.16) is Alice's original state $|\phi\rangle$ after the unitary U has been applied.

Thus, if Alice perform a measurement on her qubits using the rotated Bell basis in Eq. (2.11), Bob's state would be one of the following possibilities (depending on Alice's outcome):

Alice measurement's outcome	Bob's resulting state
00	$\alpha U\|0\rangle+\beta U\|1\rangle$
01	$\alpha U\|1\rangle+\beta U\|0\rangle$
10	$\alpha U\|0\rangle-\beta U\|1\rangle$
11	$\alpha U\|1\rangle-\beta U\|0\rangle$

Therefore, Bob is able to produce the state $U|\phi\rangle$ in his lab by applying the correction operator X^vZ^w to the qubit he possesses, where $v,w\in\{0,1\}$ are the two classical bits sent by Alice. Since an arbitrary SU(2) rotation can be decomposed as $U(\alpha,\beta,\gamma) = R_z(\gamma)R_x(\beta)R_z(\alpha)$ (for some α,β,γ), arbitrary single-qubit unitaries can be implemented by composing the teleportation protocol with rotated Bell bases, more specifically, with unitaries $U_1 = R_z(\theta)$ and $U_2 = R_x(\theta)$.

2.1.3.2 The CNOT gate

Consider the scenario where two spatially separated qubits need to be teleported to the same place. This can be achieved by simply considering a *double teleporter* (Fig 3), that is, two separated teleportation protocols where the "Bob" parts of each protocol are at the same place (or arbitrarily close to each other). A few comments about this protocol are noteworthy. First of all, remember that the teleportation protocol does not destroy

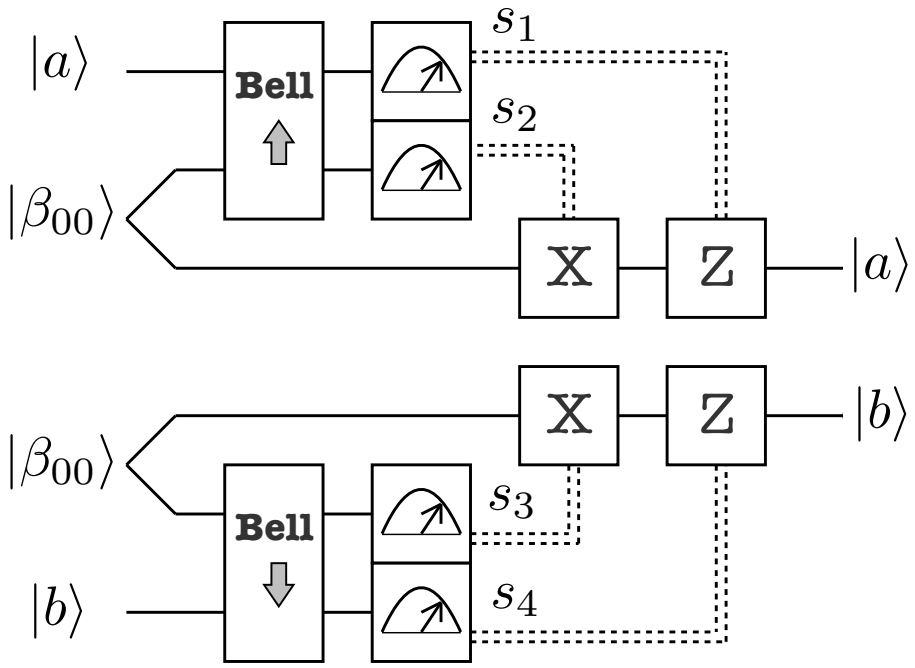


Figure 3: The double teleporter: two teleportation protocol circuits (Fig. 2) combined to teleport two spatially separated states $|a\rangle$ and $|b\rangle$ to the same place.

the correlations of the teleported qubits. It can be used to teleport both entangled and mixed states.

Without loss of generality, suppose now that the double teleporter is used simply to teleport two different single-qubit states $|a\rangle$ and $|b\rangle$. A $CNOT$ can be applied at the end of the circuit, as is done in Fig. 4, and then the computation can be carried on. The implementation of $CNOT$ gates, together with the ability of implementing arbitrary $SU(2)$ rotations (Sec. 2.1.3.1), is enough to achieve universality in quantum computing.

However, if one wants to implement a $CNOT$ gate using only resource state preparation, measurements and classically controlled operations - as is the case for single-qubit gates (Sec. 2.1.3.1) - an adaptation of the double teleporter needs to be done. Starting from Fig. 4, we move the $CNOT$ gate backwards through the Pauli gates until it reaches the beginning of the circuit. By doing so, the set of Pauli gates in the circuit changes to a different set of gates (still Pauli gates), as depicted in Fig. 5. Therefore, if the state $|\chi\rangle_{abcd} = CNOT_{bc}|\beta_{00}\rangle_{ab}|\beta_{00}\rangle_{cd}$ can be produced, a $CNOT$ gate can be applied to any two qubits using the double teleporter protocol.

The teleportation-based quantum computer uses adaptations of the quantum teleportation protocol to perform computations. Single-qubit unitaries are implemented by using the rotated Bell basis (Eq. 2.11) instead of the standard Bell basis from the original tele-

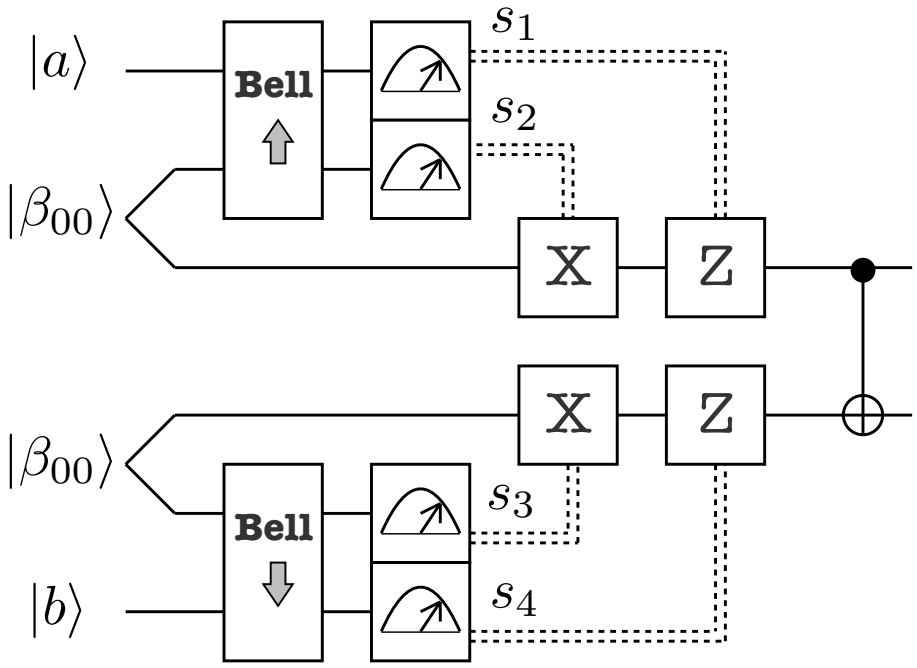


Figure 4: A double teleporter with a $CNOT$ gate being applied to the teleported states.

portation protocol. This procedure requires the production (and possibly storage) of $|\beta_{00}\rangle$ states, the ability of performing measurements in the rotated Bell basis and classically controlled Pauli gates, which application depend on the Bell measurements outcomes. On the other hand, to implement a $CNOT$ gate the state $|\chi\rangle_{abcd} = CNOT_{bc}|\beta_{00}\rangle_{ab}|\beta_{00}\rangle_{cd}$ needs to be produced to be used as resource state for the double teleporter protocol. Note that both resource states, namely $|\beta_{00}\rangle$ and $|\chi\rangle_{abcd}$, can be produced *offline*, that is, before and independently of the computation. Thus, the production of these states (remember that the $|\chi\rangle$ state can be produced by applying a $CNOT$ gate to a pair of $|\beta_{00}\rangle$ states) can be probabilistic, such that only "well-produced", high-fidelity resource states are stored to be used in the computation.

In summary, using simple adaptations of the teleportation protocol [14], Gottesman and Chuang showed [52] that quantum computation can be driven by measurements only, performed on previously and independently prepared resource states. The teleportation-based model was the first measurement-based model for quantum computing; in the following years the same key ingredients present in the model proposed by Gottesman and Chuang were used to develop several other measurement-based models [95, 11, 73, 89].

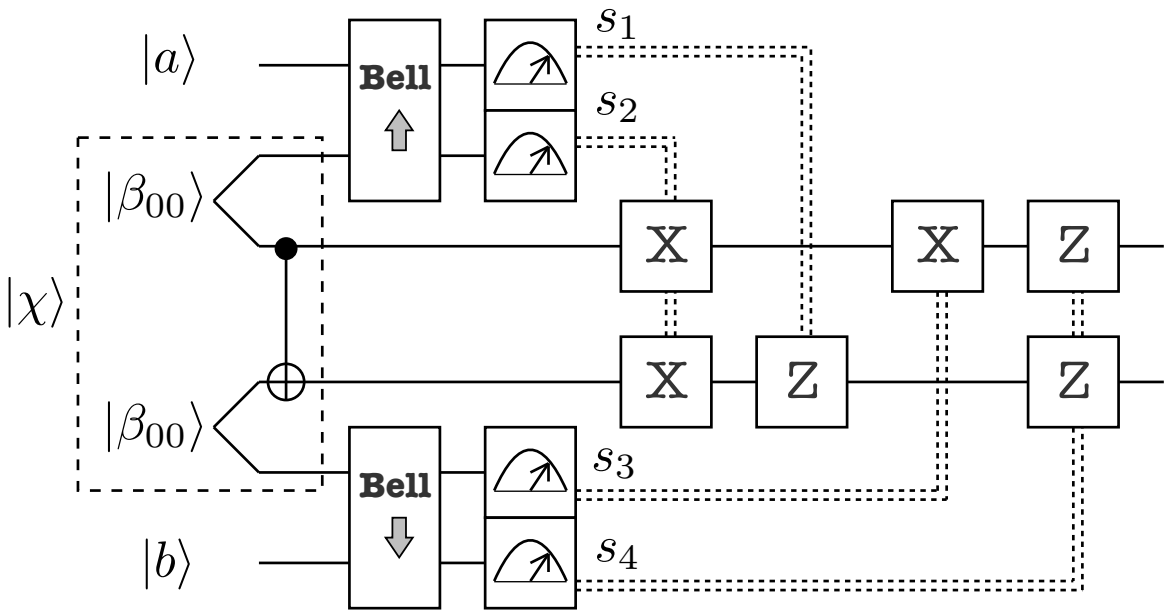


Figure 5: Adapted version of the double teleporter: using the four-qubit state $|\chi\rangle_{abcd}$, this circuit applies the $CNOT$ gate to the unknown states $|a\rangle$ and $|b\rangle$ using only measurements and classically controlled operations.

2.2 The rise of measurement-based models

In the subsequent years, several different MBQC models were proposed. Michael Nielsen [87] showed that universal quantum computing can be achieved if a quantum memory is available and it is possible to perform projective measurements on up to four qubits. In [73], Debbie Leung simplified this result showing that 2-qubit measurements are enough to achieve universality in a teleportation-based quantum computation model. Nevertheless, the model invented by Robert Raussendorf and Hans Briegel in 2001 [96], the so called *one-way quantum computation* (1WQC) model, is arguably the simplest, most elegant MBQC model.

There are at least two remarkable differences between the 1WQC model and the other MBQC models. First, entangling measurements are no longer necessary in the 1WQC model: the computation is driven by single-qubit measurements alone, in contrast with the need for entangling measurements in both Nielsen's and Leung's proposals. This simplification is only possible because the model requires the preparation of a multi-qubit highly-entangled state at the beginning, usually called the resource state, that scales with the size of the computation to be performed. Within this framework, the role of entanglement in quantum computing is highlighted; Since all entanglement needed for the

computation is generated before the actual computation starts, identifying which entangled quantum states can be used as resource states in 1WQC has become an active area of research.

A remark on terminology. Some authors refer to the one-way model as the *cluster state model*, because the cluster state was the first proven universal resource for 1WQC. In this thesis I refer to an MBQC model driven by single-qubit measurements as the one-way model, regardless of the resource state being used. This terminology is in agreement with the spirit behind the name: it is called one-way model because the resource state is "consumed" during the computation, possibly becoming completely useless after the end of the computation. In contrast, in the MBQC models where entangling measurements are allowed, even the measured qubits can be recycled and used again in the computation.

2.3 A brief description of the one-way QC model

In this section I give a brief description of the main concepts of the one-way model, which will be explained in more depth in the following sections. The one-way model for quantum computing requires the initial preparation of a multi-qubit highly entangled state over which the information processing will be driven by single-qubit measurements. A particularly simple resource state is the so-called *cluster state*, which can be constructed by arranging qubits in a square lattice and allowing an Ising-type interaction between neighboring qubits (see Fig. 6-a). The cluster state is said to be a universal resource[2] for 1WQC because any quantum algorithm can be performed using a cluster state of sufficient size.

Once the resource state is prepared, the information processing is driven by projective single-qubit measurements. Each time a measurement is performed, the measured qubit becomes unentangled with respect to the remaining qubits in the resource state; the measured qubit itself has no importance for the computation being implemented - only the measurement outcome has - and therefore can be discarded. Depending on the measurement outcome, the remaining resource state is projected onto one of two possible states - each of which associated to one measurement outcome. If the resource state is projected onto an 'undesired' state[3], a set of corrections must be applied to map it to the

[2] A formal definition of universal resources for 1WQC is given later, in Def. 1 (Sec. 2.4).

[3] At this point is hard to make clear what "undesired state" means in our context. For now, it suffices to say that this is the scenario where the computation is not heading to the correct answer (hence "undesired state"), and hence must be mapped back onto the right track. In Chapter 3 I explain in detail when and

'correct' state, before the next measurement takes place. Alternatively, the measurement bases of the next measurements can be modified in order to counteract the side-effects of previous measurements. This process continues until the algorithm is done and there are no more measurements to be performed. Let us now analyze each step of the 1WQC model more carefully:

- **Resource state.** There are several different quantum states that can be useful resources for 1WQC. There are both 'natural' - arising as the ground state of a Hamiltonian - and artificial resource states, which need to be constructed before the actual computation begins. A specially useful resource state family is the so called *graph states* [60]. Graph states can be created in two steps: (1) Preparation of n qubits in the $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ state and (2) application of the unitary gate $CZ_{ij} = diag(1, 1, 1, -1)$ to selected pairs of qubits i and j. These states are called graph states because a graph can be used to represent the physical system, by associating each qubit to a vertex and each application of a CZ gate to an edge in the graph. Moreover, there are sub-classes of graph states which are known to be universal resources for 1WQC, in the sense that arbitrary quantum computation can be performed using these states. The most well-known case of universal graph state is the square lattice graph state, usually referred to as the *cluster state*. The issue of universality of resource states will be discussed in Section 2.4.

- **Measurements.** The information processing is driven by projective measurements performed using appropriately chosen bases, which depend on the resource state over which the measurements will be performed and the computation we would like to run. For graph states, the following bases are usually used:

$$\mathcal{B}_i(\theta_i) = \left\{ \frac{|0\rangle_i + e^{i\theta_i}|1\rangle_i}{\sqrt{2}}, \frac{|0\rangle_i - e^{i\theta_i}|1\rangle_i}{\sqrt{2}} \right\}, \tag{2.17}$$

  which for the sake of simplicity will be represented simply as $\mathcal{B}_i(\theta_i) = \{|+_{\theta_i}\rangle, |-_{\theta_i}\rangle\}$. To represent the measurement result, we use the bit value $s \in \{0, 1\}$ as a label for the basis eigenvectors; Hence, $s_i = 0$ means that qubit i was projected onto the $|+_{\theta_i}\rangle$ eigenstate whereas $s_i = 1$ is associated to $|-_{\theta_i}\rangle$. The measured qubit itself is not important for the computation and can therefore be discarded after the bit value s_i is recorded. The measurement outcome tells us what is the state of the remaining

how an undesired state (obtained after a measurement have been performed) can be corrected, a problem which is intimately related to the usefulness of a given resource state.

qubits in the resource state, where the computation is actually being performed. In section 2.5 I explicitly describe the effect of measuring a graph state qubit in the $\mathcal{B}_i(\theta_i) = \{|+_{\theta_i}\rangle, |-_{\theta_i}\rangle\}$ basis and discuss the role of measurements in the Pauli bases $\theta = 0$ and $\theta = \pi/2$.

- **Adaptiveness.** Due to the randomness of the measurement results, it is not possible to predict the quantum state of the remaining qubits in the resource state. After each measurement, the resource state collapses onto one of two possible states, each of which associated to the measurement outcome $s \in \{0, 1\}$. This randomness can be accounted for by adapting the measurement angles of future measurements depending on the outcome of previous ones. This concept of canceling the side-effects of randomness on the resource state in order to perform deterministic computation - a technique known as *adaptive measurements* - will be further detailed in Section 2.6.1.

- **Output.** The output of a 1WQC protocol can be either quantum or classical. In the case where the goal is to prepare a specific quantum state, a subset of the cluster qubits is left unmeasured and some final unitary corrections must be applied to counteract the random effect of previous measurements. On the other hand, in the cases where some calculation is being performed - and hence a classical output is expected - the output qubits are commonly measured in the computational basis $\{|0\rangle, |1\rangle\}$ and the measurement results reinterpreted in the light of all previous measurement outcomes that might have influenced the read-out measurements.

A one-way quantum computer can be seen as a classical computer which has access to a quantum resource state. The classical computer has the algorithm stored in its memory - a description of the order in which measurements must be done and the corresponding measurement angles - as well as the results of all performed measurements. Moreover, the classical computer needs to calculate whether a given measurement angle must be adapted to counteract the randomness of previous measurements, changing the measurement angle if necessary. In fact, this description of a one-way quantum computer makes even more evident one of the main properties of the model: the clear separation between classical and quantum computational resources. In contrast to the quantum circuit model, where the whole information processing is driven by quantum operations, in the one-way quantum

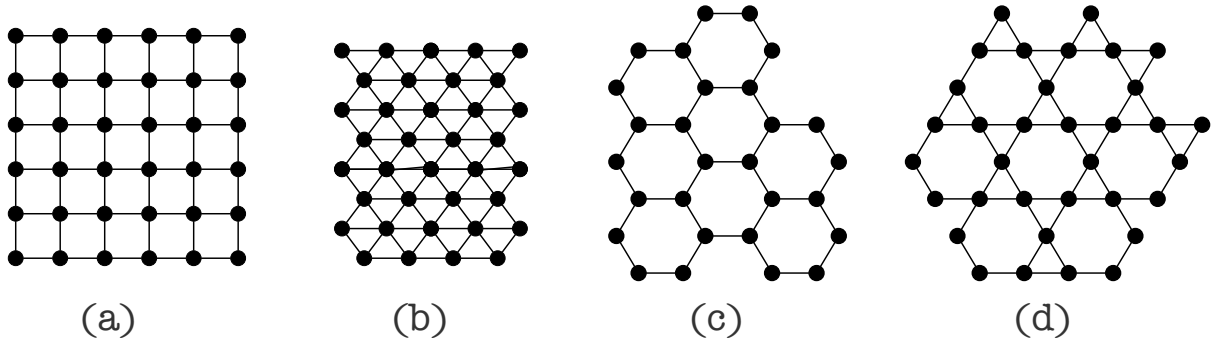


Figure 6: Universal graph states: (a) Square Lattice, (b) Triangular Lattice, (c) Hexagonal Lattice, (d) Kagome Lattice.

computer all 'quantumness' is in the resource state; the information processing is driven by calculating some functions of the correlations between the measurements performed over the resource state.

2.4 Resource states

A resource state is a multi-qubit highly entangled state used as a resource for a one-way quantum computer. In the 1WQC model, all entanglement that will be used in the computation is available right at the beginning - no entangling operation needs to be applied during the computation. The resource state can be prepared in several different configurations, depending on the computational task to be performed. There are some resource state families which enable any computation to be performed. We refer to those families of states as *universal resource state families* for 1WQC. More formally, universal resource state families can be defined as follow:

Definition 1 ***(Universal resource state families [108])*** *Let Ψ be a family of infinitely many states: $\Psi = \{|\psi_1\rangle, |\psi_2\rangle, ...\}$. The family of states Ψ is said to be a universal resource state family for 1WQC if for each n-qubit state $|\phi\rangle$ there exists a m-qubit ($m \geq n$) state $|\psi\rangle \in \Psi$ such that the transformation $|\psi\rangle \rightarrow |\phi\rangle|+\rangle^{m-n}$ can be implemented deterministically by local operations and classical communication (LOCC).*

That is, any quantum state $|\phi\rangle$ can be prepared using only states within the family Ψ. In other words, any unitary operation U such that $|\phi\rangle = U|+\rangle^n$ can be implemented. Note that here universality is a property of a family of states, and not of a single state[4].

[4]For the sake of simplicity, however, we usually say that cluster states, for instance, are a universal resource for 1WQC, although the precise statement would be that the cluster state family is a universal resource state family for 1WQC.

The one-way model highlights the role of entanglement in quantum computing. Measurements on entangled states never increase the amount of entanglement and therefore, in the one-way model the entanglement is "consumed" during the computation. Initially, it was believed that a higher amount of entanglement would result in a greater usefulness of the quantum state. However, Gross, Flammia and Eisert [56] showed that some quantum states are too entangled to be useful, *i.e.*, that the correlation in the statistics of any set of measurements over such states could be classically simulated. On the other hand, several states are known not to have enough entanglement to be useful for 1WQC, as it is the case for the 1-dimensional cluster state [88] and rectangular cluster states having one side that scales logarithmically with respect to the other one [114].

A resource state can be artificially constructed or it can be a "natural" state, arising as the ground state of a given Hamiltonian. A good example is the Affleck-Kennedy-Lieb-Tasaki (AKLT) state [8], that first appeared in the context of condensed matter physics. The AKLT state is the ground state of a two-body interaction Hamiltonian and it is separated from the excited states by a finite energy gap. It was shown that 1-dimensional AKLT chains can be used to perform simple quantum computing tasks [83] and that by properly coupling many AKLT chains universal quantum computing can be achieved [19]. In [113], it was shown that two-dimensional AKLT state on a honeycomb lattice is also a universal resource for measurement-based QC. Thus, AKLT states stand as one of the most interesting resource states for 1WQC, due to the possibility of preparing it in solid state systems just by cooling them.

Although two-dimensional cluster states can be used as a resource for universal quantum computation in the one-way model, arbitrary graph states may, or may not, serve for the same purpose; investigating which kinds of entangled states are useful resources for 1WQC is an important and active area of research [56, 108, 85, 112].

2.4.1 Cluster states and graph states

A particularly useful class of resource states are the so-called *graph states* [60]. A graph state is the state of a multiqubit system that can be represented by a graph in the following way: Each qubit of the system is represented by a vertex and the entanglement between two qubits is represented by an edge. More formally, graph states can be defined as follows.

Definition 2 ***(Graph states [60]).*** *Let $G = (V, E)$ be a graph, where V and E are the*

sets of vertices and edges of G, respectively. A graph state $|G\rangle$ is the unique simultaneous eigenstate with eigenvalue $+1$ of the operators:

$$K_i = X_i \prod_{j \in N(i)} Z_j, \quad \forall i \in V \tag{2.18}$$

where $N(i)$ stands for the set of vertices which are neighbors of i in G.

Cluster states are a particular type of graph states, where the graph is a square lattice of some dimension d, as shown in Figure 6-a. Due to the restricted entanglement geometry of cluster states, a given algorithm can in general be implemented using fewer qubits in a graph state with more suitable geometry. However, there are several physical implementations where the regular design of the cluster make it easier to be constructed.

Proposition 1 ***(Graph state preparation [100]).*** *A graph state can be created by preparing a collection of qubits in the $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ state and letting them evolve under the Hamiltonian:*

$$\mathcal{H} = \hbar g \sum_{(i,j) \in E_G} \frac{I^{(i)} - \sigma_z^{(i)}}{\sqrt{2}} \otimes \frac{I^{(j)} - \sigma_z^{(j)}}{\sqrt{2}} = \hbar g \sum_{(i,j) \in E_G} |1\rangle_i\langle 1| \otimes |1\rangle_j\langle 1| \tag{2.19}$$

for a time $T = \frac{\pi}{g}$.

Proof. The unitary evolution driven by the Hamiltonian in Eq. (2.19) can be written as:

$$U_{\mathcal{H}} = exp(i\frac{T\mathcal{H}}{\hbar}) = exp\Big(i\pi \sum_{(i,j) \in E_G} |1\rangle_i\langle 1| \otimes |1\rangle_j\langle 1|\Big)_{ij} = \prod_{(i,j) \in E_G} exp\Big(i\pi |11\rangle_{i,j}\langle 11|_{ij}\Big) \tag{2.20}$$

This is a product of the so-called *controlled-phase* gate, or simply CZ gate, and can be more conviently written as:

$$CZ_{ij} = |0\rangle_i\langle 0| \otimes I_j + |1\rangle_i\langle 1| \otimes Z_j \tag{2.21}$$

which applies the Pauli Z operator (phase flip) to qubit j if qubit i is in state $|1\rangle$. The CZ gate satisfies the following properties:

$$CZ_{ij}(X_i)CZ_{ij}^{\dagger} = X_iZ_j, \tag{2.22}$$

$$CZ_{ij}(X_j)CZ_{ij}^{\dagger} = Z_iX_j, \tag{2.23}$$

$$CZ_{ij}(Z_i)CZ_{ij}^{\dagger} = Z_i, \tag{2.24}$$

$$CZ_{ij}(Z_j)CZ_{ij}^{\dagger} = Z_j. \tag{2.25}$$

Now we are in the position to prove the proposition. Evolving the multi-qubit (separable) state $|+\rangle^{\otimes n}$ accordingly with $U_{\mathcal{H}}$ yields:

$$U_{\mathcal{H}} \otimes_{i\in V_G} |+\rangle = U_{\mathcal{H}} \prod_{u\in V_G} X_u \bigotimes_{i\in V_G} |+\rangle \tag{2.26}$$

$$= \prod_{(i,j)\in E_G} CZ_{ij} \prod_{u\in V_G} X_u \bigotimes_{i\in V_G} |+\rangle \tag{2.27}$$

$$= \prod_{u\in V_G} \Big(X_u \prod_{v\in N(u)} Z_v\Big) \prod_{(i,j)\in E_G} CZ_{ij} \bigotimes_{i\in V_G} |+\rangle \tag{2.28}$$

$$= \prod_{u\in V_G} K_u|G\rangle \tag{2.29}$$

where Eq. (2.18) was used in the last equality. Since Eq. (2.29) satisfies Definition 2, the statement in this proposition holds.□

Let us now consider as an example the 1D cluster state with three qubits. The graph state associated to this system can be constructed as follows:

$$|G_3\rangle = CZ_{12}CZ_{23}|+\rangle_1|+\rangle_2|+\rangle_3 \tag{2.30}$$

$$= \frac{1}{2}CZ_{12}|+\rangle_1\Big(|00\rangle_{23} + |01\rangle_{23} + |10\rangle_{23} - |11\rangle_{23}\Big) \tag{2.31}$$

$$= \frac{1}{2\sqrt{2}}\Big(|000\rangle + |100\rangle + |010\rangle - |110\rangle \tag{2.32}$$

$$+ \quad |001\rangle + |101\rangle - |011\rangle + |111\rangle\Big)_{123} \tag{2.33}$$

It is easy to verify that this state satisfies the three eigenvalue equations:

$$X_1 Z_2 |G_3\rangle = |G_3\rangle \quad (2.34)$$

$$Z_1 X_2 Z_3 |G_3\rangle = |G_3\rangle \quad (2.35)$$

$$Z_2 X_3 |G_3\rangle = |G_3\rangle \quad (2.36)$$

as required by definition 2. The graph state we just constructed is locally equivalent to the well-known GHZ state [2], which is widely used in several quantum information protocols. To see this, first apply the Hadamard gate to qubit 2 and then change the basis from $\{|0\rangle, |1\rangle\}$ to $\{|+\rangle, |-\rangle\}$:

$$I_1 \otimes H_2 \otimes I_3 |G_3\rangle = \frac{1}{2}\Big(|000\rangle + |110\rangle + |011\rangle + |101\rangle\Big) \quad (2.37)$$

$$= \frac{1}{\sqrt{2}}\Big(|++{+}\rangle + |--{-}\rangle\Big) \quad (2.38)$$

which is the GHZ state.

The first universal resource to be discovered was the two-dimensional cluster state. Following this result, other 2D regular lattices were proven to be universal resources, such as the triangular, hexagonal and Kagome lattices [108] (see Fig. 6). Moreover, any graph from which one can obtain (by deleting some vertices) one of the aforementioned universal lattices can also be considered as a universal resource (this is the case for some fractal lattices [78]). This is so because a qubit can be easily removed from a graph state by simply measuring it in the computational basis $\{|0\rangle, |1\rangle\}$, as we show in Section 2.5.2.

A non-universal graph state is not necessarily useless for one-way quantum computing. There might exist some specific computational tasks for which non-universal graph states can be used and still outperform classical computers. It is an active area of research to identify which resource states are useful for 1WQC as well as which lead to measurement statistics that can be efficiently simulated in a classical computer.

2.5 Measuring graph state qubits

In the last section we reviewed several different quantum states that can be used for quantum information processing in the 1WQC model. Once the resource state has been prepared, the quantum information processing is driven by *projective measurements*:

Definition 3 ***(Projective measurements [86]).*** *A projective measurement is described by an observable M, which is an Hermitian operator with spectral decomposition given by*

$$M = \sum_m m P_m \tag{2.39}$$

where P_m is the projector onto the eigenspace of M associated to the eigenvalue m. The possible outcomes of the measurement are labelled by the eigenvalues of observable M. If a quantum system in the $|\psi\rangle$ state is measured, the probability of getting the outcome m is given by

$$p(m) = \langle\psi|P_m|\psi\rangle, \tag{2.40}$$

whereas the resulting state (in the case outcome m occurred) would be

$$\frac{1}{\sqrt{p(m)}} P_m|\psi\rangle. \tag{2.41}$$

Moreover, the projection operators $\{P_m\}$ must satisfy the completeness relation, $\sum_m P_m = I$, and the orthogonality relation, $P_m P_{m'} = \delta_{mm'} P_m$.

Two remarks about nomenclature shall be made at this point. Most times I will refer to the set of orthogonal projectors $\{P_m\}$, instead of the associated observable M, to describe a projective measurement. Moreover, the phrase "measurement in the basis $|m\rangle$" (where $\{|m\rangle\}$ denotes an orthonormal basis) refers to the measurement of observable M, constructed according to Eq. (2.39) using the projectors $P_m = |m\rangle\langle m|$.

A simple but important example is the single-qubit measurement in the $\{|0\rangle, |1\rangle\}$ basis (*i.e.*, the Pauli Z observable), which in quantum information is known as the *computational basis*. The projectors are $P_0 = |0\rangle\langle 0|$, associated with the eigenvalue $m = 0$ and $P_1 = |1\rangle\langle 1|$, associated to $m = 1$. Suppose that the qubit is going to be measured is in the $|\psi\rangle = a|0\rangle + b|1\rangle$ state. Then, the probability of obtaining the measurement outcome $m = 0$ is

$$p(0) = \langle\psi||0\rangle\langle 0||\psi\rangle = |a|^2. \tag{2.42}$$

Similarly, the probability of obtaining the measurement result $m = 1$ is given by $p(1) = |b|^2$.

In Section 2.5.2 we will see how measurement in the computational basis plays an important role in computing using graph states. Before getting to that, however, I shall introduce the measurement bases usually used in the 1WQC model (Section 2.5.1). Last but not least, I analyze in Section 2.5.3 the role of measurement in the $\{|+\rangle, |-\rangle\}$ basis[5],

[5]That is, the measurement of the Pauli X observable.

that can be used to teleport logical qubits from one point to another in the graph.

2.5.1 Single-qubit measurements

Let us start by analyzing the effect of a single-qubit measurement in a simple graph state. For reasons that will become clear later, we use the measurement bases in Eq. (2.17), reproduced below:

$$\mathcal{B}_i(\theta_i) = \left\{ \frac{|0\rangle_i + e^{i\theta_i}|1\rangle_i}{\sqrt{2}}, \frac{|0\rangle_i - e^{i\theta_i}|1\rangle_i}{\sqrt{2}} \right\}, \tag{2.43}$$

which can be represented simply as $\mathcal{B}_i(\theta_i) = \{|+_{\theta_i}\rangle, |-_{\theta_i}\rangle\}$. Consider an input state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and a resource state $|G\rangle$ prepared in the following way:

$$\begin{aligned} |G\rangle &= CZ_{12}|\psi\rangle_1|+\rangle_2 && (2.44)\\ &= CZ_{12}\big(\alpha|0\rangle + \beta|1\rangle\big)_1\Big[\frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)\Big]_2 && (2.45)\\ &= \frac{1}{\sqrt{2}}\big(\alpha|00\rangle + \alpha|01\rangle + \beta|10\rangle - \beta|11\rangle\big)_{12} && (2.46) \end{aligned}$$

Measuring the first qubit in the basis $\mathcal{B}_i(\theta_i) = \{|+_{\theta_i}\rangle, |-_{\theta_i}\rangle\}$, we obtain one out of two possibilities. If qubit 1 is projected onto the $|+_\theta\rangle$ state, the eigenstate associated to the $+1$ eigenvalue, qubit 2 is projected into the state $\frac{1}{\sqrt{2}}\Big[\big(\alpha + e^{-i\theta}\beta\big)|0\rangle + \big(\alpha - e^{-i\theta}\beta\big)|1\rangle\Big]$. On the other hand, if the measurement result is $|-_\theta\rangle$, qubit 2 is projected into the state $\frac{1}{\sqrt{2}}\Big[\big(\alpha - e^{-i\theta}\beta\big)|0\rangle + \big(\alpha + e^{-i\theta}\beta\big)|1\rangle\Big]$. Thus, the state of qubit 2 can be represented as

$$X^s HP(\theta)|\psi\rangle, \tag{2.47}$$

where X is the Pauli operator, s is a bit $s \in \{0, 1\}$, H is the Hadamard gate and $P(\theta) = diag(1, e^{i\theta})$ the phase gate. By definition, $s = 0$ is associated to the eigenstate $|+_\theta\rangle$ while $s = 1$ gives the eigenstate $|-_\theta\rangle$. The Pauli operator X^s, which depends on the measurement result s, is called the *by-product* operator. In fact, by-product operators can be composed by Pauli operators X and Z; they appear in the computation as a consequence of the randomness of the results of individual measurements[6]. The simple example above shows how to teleport a quantum state through the gate:

$$J(\theta) = HP(\theta) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & e^{i\theta} \\ 1 & -e^{i\theta} \end{pmatrix} \tag{2.48}$$

[6] As we will see in Sec. 2.6.1, those by-product operators become troublesome when composing several J gate teleportation protocols. We will see that they can be accounted for by adapting measurement bases during the process of computation, according to the results of previously performed measurements.

using only a two-qubit graph state, one single-qubit measurement and a Pauli correction. From now on, I will refer to the aforementioned protocol as the *J-gate teleportation* protocol. As I explain in section 2.6.3, $J(\theta)$ gates play a central role in this model since together with the CZ gate they constitute a universal gate-set for quantum computing.

2.5.2 Removing redundant qubits

When using universal graph states (*e.g.*, a cluster state) to implement a 1WQC protocol that simulates several concatenated gates, it might happen that some qubits in your resource state must be removed. For instance, suppose one wants to simulate the implementation of a $CNOT$ gate as in Figure 7-b but using a 3×3 cluster state; five qubits would need to be removed in order to have the exact graph depicted in Figure 7-b. Fortunately, qubits can be easily removed from graph states by simply measuring them in the Z eigenbasis. This follows from the definition of graph states (Def. 2) and the pair of identities:

$$|0\rangle_i\langle 0|CZ_{ij} = |0\rangle_i\langle 0| \otimes I_j \quad (2.49)$$

$$|1\rangle_i\langle 1|CZ_{ij} = |1\rangle_i\langle 1| \otimes Z_j \quad (2.50)$$

Therefore, if the measurement outcome is $s_i = 0$, there is no effect on the remaining qubits in the graph (Eq. 2.49); if $s_i = 1$, all qubits initially connected by an edge to the measured qubit will gain a phase (Eq. 2.50), given by the Pauli Z gate. As will be shown in Section 2.6.1, Pauli corrections can be handled by simply adapting the measurement basis on that site. Therefore, the side effect of measuring a qubit in the Z basis can be easily compensated.

Interestingly, measurements in the X and Y eigenbasis also result in new graph states: they not only delete a vertex from the graph (as the Z measurement does) but also change the geometry of the graph nearby the measured qubit. An overview of the effect of Pauli measurements in graph states can be found in Appendix 1.

2.5.3 Moving the logical state of a qubit in a graph

Another particularly useful measurement basis is the obtained by fixing $\theta = 0$ in Eq. (2.17), *i.e.*, measuring the Pauli X observable. Suppose the one needs to move a qubit state from one part to another in the graph (similarly to the teleportation protocol). There are several situations where that might be necessary (or at least convenient), as

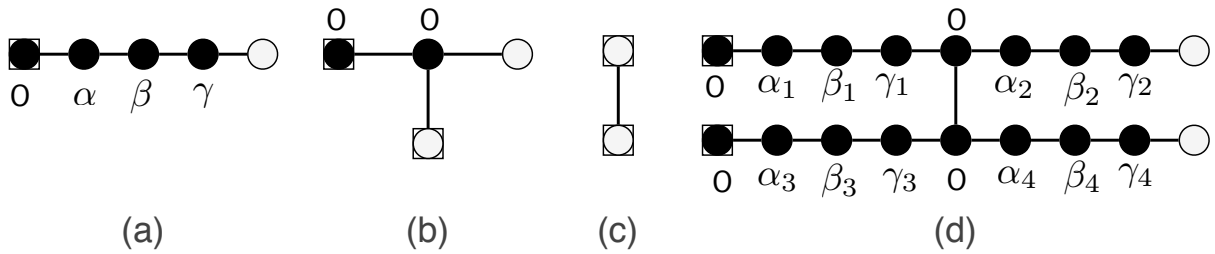


Figure 7: Graphs implementing (a) Arbitrary single-qubit unitary, (b) $CNOT$ gate, (c) CZ gate, (d) two arbitrary single-qubit rotations followed by a CZ gate and then other two single-qubit rotations.

for example in a graph with such a geometry that a two-qubit gate can be simulated just in specific regions of that graph. Hence, we would frequently need to move qubit states around so that two-qubit gates can be applied to any pair of them.

Suppose we need to move a qubit state (in state, say, $|\psi\rangle$) from a given vertex i to another j. First, we need to build a "wire" with an even number of qubits linking the two vertices; this can be done by deleting the qubits in the neighborhood of the wire (using measurements in the Z basis), resulting in a wire of qubits connected by edges as depicted in Fig. 7-a, for instance. Then, we measure all qubits in the wire in the X basis. Each measurement will teleport the state to the next vertex while applying the gate $J(\theta = 0) = H$, as explained in Section 2.5.1. Since $H^N = 1$ (for N even), by the end of the process we will have moved the qubit state to vertex j, as desired.

2.6 Simulating the circuit model

The 1WQC model is equivalent to other quantum computing models. For instance, arbitrary quantum circuits can be efficiently simulated using the one-way model if a graph state of sufficient size is available and arbitrary single-qubit measurements can be performed. In this section I explicitly show: (i) how to simulate arbitrary quantum circuits. To do so, it is sufficient to show how to simulate any gate of an universal gate-set; and (ii) that the subgraphs simulating each of those gates can be "glued" together, allowing the concatenation of arbitrary gates.

It turns out that the gate simulated by measuring a graph state qubit in the $\mathcal{B}(\theta)$ bases (as in the J gate teleportation shown in Section 2.5.1), namely $J(\theta) = HP(\theta)$, together with the CZ gate, constitute an universal gate-set for quantum computing [34]. In the following sections I show how to concatenate many J-gate teleportation protocols in order to simulate any given quantum circuit.

2.6.1 Universal single-qubit rotations and the need for adaptive measurements

Here I show how to compose the J-gate teleportation protocol shown in Section 2.5.1. As an useful example, we show how to perform arbitrary single-qubit $SU(2)$ rotations. Using the Euler decomposition, we can write such unitary as

$$U(\alpha, \beta, \gamma) = R_z(\gamma)R_x(\beta)R_z(\alpha) \tag{2.51}$$

for some angles γ, β, α. We can rewrite Eq. (2.51) in terms of Eq. (2.48) (the J gate) by using the identities $R_z(\phi) = J(0)J(\phi)$ and $R_x(\phi) = J(\phi)J(0)$. Thus, an arbitrary rotation $U \in SU(2)$ can be written using only J gates:

$$U(\alpha, \beta, \gamma) = J(0)J(\gamma)J(\beta)J(\alpha). \tag{2.52}$$

In the previous section, we have seen how to apply a J gate to an arbitrary state $|\psi\rangle$ using only single-qubit measurements on a Bell pair and classical processing. The question that naturally arises is the following: Could one apply the unitary in Eq. (2.52) by composing the J gate teleportation protocol several times? This is indeed possible, but it involves some issues that we now analyze. First, consider the following procedure. Apply the J gate teleportation protocol to a state $|Input_1\rangle = |\psi_0\rangle$ but instead of measuring the output qubit in the computational basis (and doing some classical post-processing to account for the X^s factor), we feed it to the protocol again, resulting in the output state $|Output_2\rangle = X^{s_2}J(\theta_2)X^{s_1}J(\theta_1)|\psi_0\rangle$. Repeating this process two more times, we end up with the following description of the final state:

$$X^{s_4}J(0)X^{s_3}J(\theta_3)X^{s_2}J(\theta_2)X^{s_1}J(\theta_1)|\psi_0\rangle \tag{2.53}$$

which means that the desired unitary implementation occurs only when $s_1 = s_2 = s_3 = 0$ (the X^{s_4} can be accounted for by reinterpreting the read-out measurement, performed in the computational basis). Therefore, the X^s operators in Eq. (2.53) (known as *by-product* operators), which can easily be dealt with when performing a single application of the J gate teleportation protocol, become a troublesome factor when several of those protocols are composed.

One could think about solving this problem by evolving the output qubit of each J gate teleportation protocol (whenever $s = 1$) according to to a Hamiltonian that implements a X gate, before feeding it as the input of the next protocol. By doing so, the final state

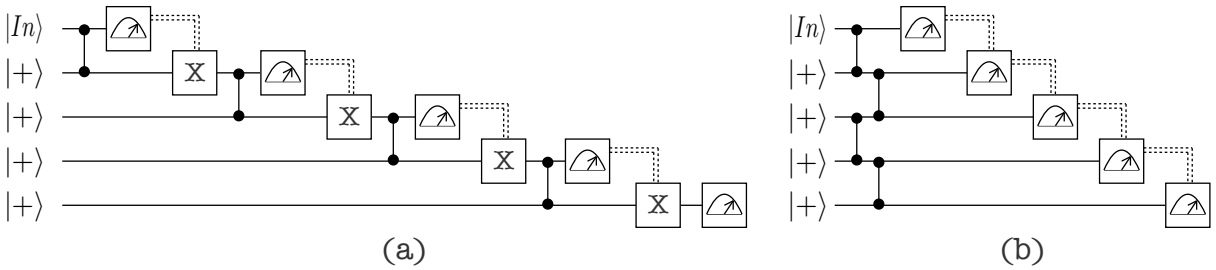


Figure 8: (a) composed J gate teleportation protocol and (b) 1WQC protocol using adaptive measurements to implement the same unitary as in (a).

would be:

$$X^{s_4} J(0)J(\theta_3)J(\theta_2)J(\theta_1)|\psi_0\rangle \tag{2.54}$$

which applies the desired unitary. Although this procedure gives the expected result, it can no longer be considered a 1WQC protocol. In one-way quantum computation, the information processing is driven by single-qubit measurements only and therefore unitary evolutions (to apply X) are not allowed during the computation. Interestingly, a genuine 1WQC protocol can be constructed from the "composed protocol" above, if we allow some measurement angles to be conditioned to previous measurements outcomes. In what follows I show how to use this conditioned choice of measurement angles to cancel out the by-product operators of previous measurements; this technique is usually referred to as *adaptive measurements*.

First, note that the measurement on qubit i commutes with gate CZ_{jk} for $j, k \neq i$. Therefore, as long as the measurements are performed in sequence (from qubit 1 to 4), all CZ gates used in the composed protocol can be applied before any measurement takes place. By doing so, a 5-qubit entangled state is prepared before the computation begins, as required in the 1WQC model. Next, we need to remove the X^s operators from in-between the J gates in Eq. (2.53). Using the the identities $J(\theta)X = ZJ(-\theta)$ and $J(\theta)Z = XJ(\theta)$, Eq. (2.53) can be rewritten as

$$X^{s_2+s_4} Z^{s_1+s_3} J(0)J[(-1)^{s_2}\theta_3]J[(-1)^{s_1}\theta_2]J(\theta_1)|\psi_0\rangle \tag{2.55}$$

which is no longer the implementation of a deterministic unitary, since only if $s_1 = s_2 = 0$ is the implemented unitary the desired one. This can be dealt with by allowing *adaptive measurements*, that is, the ability to change the angles depending on previous measurement outcomes:

$$\theta_1 = \alpha, \qquad \theta_2 = (-1)^{s_1}\beta, \qquad \theta_3 = (-1)^{s_2}\gamma \tag{2.56}$$

Thus, combining Eqs. (2.55) and (2.56) we have exactly Eq. (2.51), as desired. Therefore, adaptive measurements can be used to deal with the by-product operators in the middle of a computation, allowing one to apply a given unitary deterministically. This subject will be further explored in Chapter 3, where I address the question of which graph states allows for deterministic 1WQC.

2.6.2 Two-qubit gates

The simulation of a $CNOT$ gate can be implemented using the simple 4-qubit graph shown in Figure 7-b. Consider the following labels for the graph in Figure 7-b: from left to right, the top vertices are labelled by numbers 1, 2 and 3; the bottom vertex is labelled by number 4. Let qubits 1 and 4 encode the control and target input qubits respectively, while qubits 3 and 4 encode the control and target output qubits, respectively. Note that in this compact implementation of the $CNOT$ gate, qubit 4 encodes both the input target qubit (before the measurements) and the output target qubit (after measurements). Let $|\psi_1\rangle = a|0\rangle + b|1\rangle$ be the initial state of the target qubit and $|\psi_2\rangle = c|0\rangle + d|1\rangle$ the initial state of the control qubit. The graph state is constructed in the following way:

$$
\begin{aligned}
|G_T\rangle &= CZ_{24}CZ_{23}CZ_{12}|\psi_1\rangle_1|+\rangle_2|+\rangle_3|\psi_2\rangle_4 && (2.57)\\
&= CZ_{24}CZ_{23}\big(a|0\rangle_1|+\rangle_2 + b|1\rangle_1|-\rangle_2\big)|+\rangle_3|\psi_2\rangle_4 && (2.58)\\
&= CZ_{24}\Big[a|0\rangle_1\big(|0\rangle_2|+\rangle_3 + |1\rangle_2|-\rangle_3\big) + b|1\rangle_1\big(|0\rangle_2|+\rangle_3 - |1\rangle_2|-\rangle_3\big)\Big]|\psi_2\rangle_4 && (2.59)\\
&= a|0\rangle_1\Big[|0\rangle_2|+\rangle_3\big(c|0\rangle + d|1\rangle\big)_4 + |1\rangle_2|-\rangle_3\big(c|0\rangle - d|1\rangle\big)_4\Big] && (2.60)\\
&+ b|1\rangle_1\Big[|0\rangle_2|+\rangle_3\big(c|0\rangle + d|1\rangle\big)_4 - |1\rangle_2|-\rangle_3\big(c|0\rangle - d|1\rangle\big)_4\Big] && (2.61)
\end{aligned}
$$

The protocol consists in measuring qubits 1 and 2 on the $|\pm\rangle$ basis and correcting qubits 3 and 4 in accordance with the measurement outcomes $s_1, s_2 \in \{0, 1\}$. For instance, if $s_1 = s_2 = 0$, which means that qubits 1 and 2 collapsed onto the $|+\rangle$ state, the unmeasured qubits are in the joint state

$$
\begin{aligned}
|\psi\rangle_{34} &= \langle+|_1\langle+|_2|G_T\rangle_{1234} && (2.62)\\
&= a\Big(|+\rangle_3|\psi_2\rangle_4 + |+\rangle_3 Z_4|\psi_2\rangle_4\Big) + b\Big(|+\rangle_3|\psi_2\rangle_4 - |+\rangle_3 Z_4|\psi_2\rangle_4\Big) && (2.63)\\
&= ac|00\rangle_{34} + ad|11\rangle_{34} + bc|10\rangle_{34} + bd|01\rangle_{34} && (2.64)\\
&= CNOT_{43}|\psi_2\rangle_4|\psi_1\rangle_3, && (2.65)
\end{aligned}
$$

where the normalization factors were omitted. Note that the information on the target qubit has been transferred from physical qubit 1 to 3 while the control input and output

state is encoded in the same physical qubit, namely qubit 4. Analyzing the other three possible measurement outcomes $(s_1, s_2) \in \{(0,1),(1,0),(1,1)\}$, it is easy to check that the output state $|\psi_{out}\rangle$ relates to the initial state $|\psi_{in}\rangle = |\psi_1\rangle_t|\psi_2\rangle_c$ via the following transformation:

$$|\psi_{out}\rangle = X_t^{s_2} Z_t^{s_1} Z_c^{s_1} CNOT|\psi_{in}\rangle \tag{2.66}$$

which applies a $CNOT$ gate to the input state once the Pauli corrections are applied.

Since the entangling gate in the 1WQC framework we are using is the CZ gate, the simulation of a CZ gate comes "for free". In the preparation of the graph state, the CZ gate is applied to several pairs of qubits (the choice of pairs depends on the geometry of the graph) and are represented as edges in the graph. It turns out that those edges can be used at any point of the computation to simulate the application of a CZ gate to the qubit states in the vertices connected by the edges. In Chapter 3 I explain the difference between using a graph edge to implement a CZ gate or a J gate teleportation protocol.

Consider the graph in Figure 7-c, where qubits 1 and 2 are input and output qubits at the same time. It is trivial to see that this simple graph applies the CZ gate to the input state. In fact, that graph can be composed with graphs implementing other unitaries. See for instance Figure 7-d where the composition of the graph implementing the CZ gate (Fig. 7-c) is composed with four graphs able to implement arbitrary single-qubit rotations (Fig. 7-a).

2.6.3 Simulating arbitrary quantum circuits

The one way model is an universal model for quantum computation. In the last sections I have shown how to simulate both arbitrary single-qubit rotations and the $CNOT$ gate, which combined together constitute an universal gate-set for quantum computing. Moreover, the simple gate in Eq. (2.48), that can be implemented (up to a Pauli correction) using only one single-qubit measurement, together with the CZ gate also constitute a universal gate-set [34].

As an example, Figure 9 shows how a circuit is simulated in a cluster state. In the figure, the input state is encoded in the qubits at the left-most column and the output in the right-most one. First, the qubits that will not be used to simulate any gate from the circuit are removed from the cluster. This can be done by measuring them in the Z basis (as explained in Section 2.5.2); the removed qubits are represented as black dots in the figure. After that, the remaining graph has the right geometry to simulate each of the

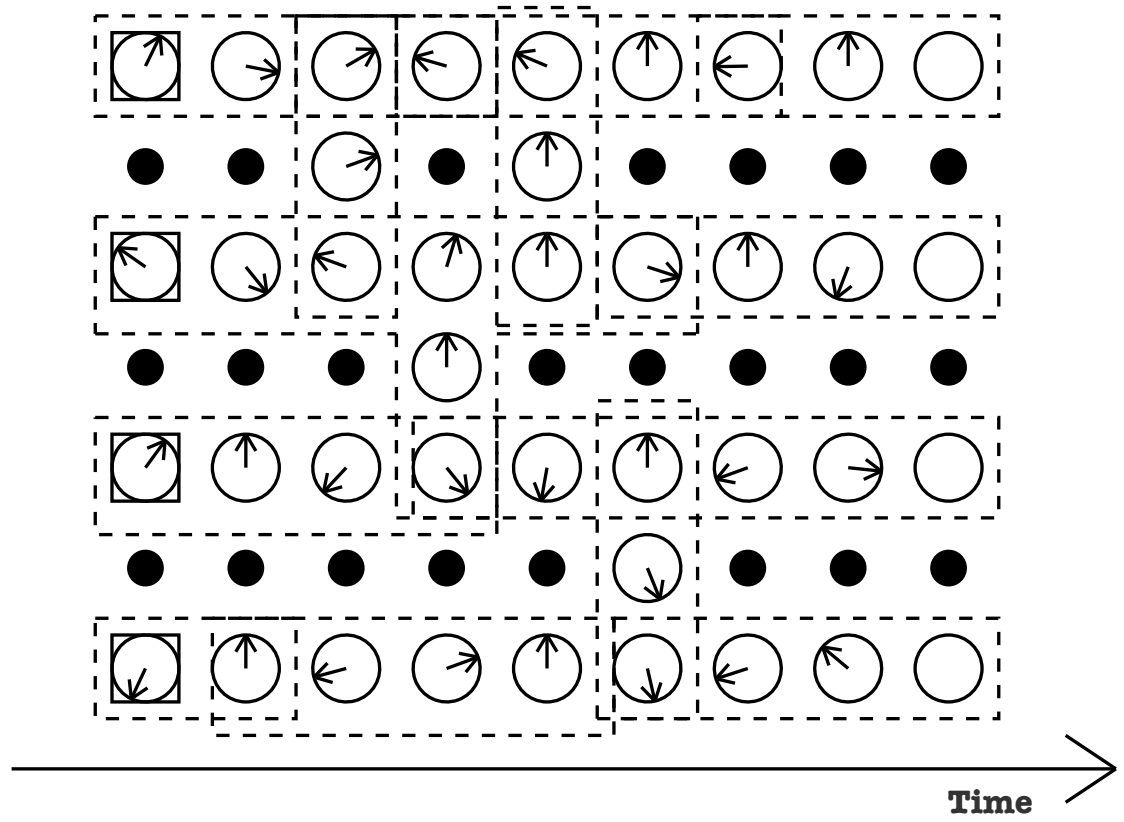


Figure 9: The simulation of a quantum circuit in a cluster state. The black dots represent qubits measured in the Z basis (and hence deleted from the cluster) and the circles with arrows represent qubits measured in the $\mathcal{B}(\theta)$ basis. Different arrow directions indicate different angles of measurement. Each set of qubits boxed with a dashed line represents the simulation of a given circuit gate.

gates from the circuit that will be simulated. There are four horizontal wires (simulating the circuit wires) and some vertical lines connecting them (simulating a two-qubit gate between the wires). Moreover, some pairs of qubits are X measured to move the output state of a gate simulator building block to be the input state of the next one, as explained in Section 2.5.3.

In fact, the universality of a measurement-based quantum computing model depends basically on two properties: The resource state and the measurement bases. When a resource state is said to be universal for QC, it means that there exist measurement bases such that the statement holds. For instance, some of the known universal graph geometries are: cluster state ($X-Y$ plane measurements + Z measurements; see Fig. 6-a) [94], Brickwork state ($X-Y$ plane measurements only) [22], Hexagonal lattice ($X-Y$ plane measurements + Z measurements; see Fig. 6-c) [108], triangular lattice ($X-Z$ plane measurements only; see Fig. 6-b) [82].

2.6.4 Beyond quantum circuit simulations

In the last sections I showed how the cluster state can be used as a 'canvas' over which a quantum circuit is simulated: after deleting some specific qubits from the cluster, the resulting graph geometry is nothing but a composition of small building blocks, each of which simulates a logical gate. However, the one-way model is not just a quantum circuit simulator. In fact, many quantum algorithms can be implemented in a more compact way when the one-way quantum computer is *not* simulating quantum circuits.

The circuit model encompasses the Schrodinger picture of Quantum Mechanics where the quantum state changes by the action of well-defined unitary evolutions. A logical gate in the circuit model is nothing but a unitary transformation in the Schrodinger picture of quantum mechanics. On the other hand, in the one-way model there is no notion of gates being applied nor unitary evolutions acting over a physical system. Therefore, the one-way model introduces a completely different way of thinking about computation when compared to the circuit model, since no logical gate is actually required (although it can be understood using gates).

For instance, there are some optimization techniques in the one-way model which are able to reduce the number of computational steps needed to simulate a quantum circuit in such a way that the 'gate building blocks' can no longer be identified. That is, although the new algorithm still implements the same computation, it can no longer be understood as a concatenation of small logical gates (as in the circuit model). The one-to-one correspondence between measurements performed and single-qubit gates implemented shown in previous examples (Section 2.6.1) is not a property shared by every one-way quantum computation. Nevertheless, to design algorithms which are not built by composing simple logic gates may represent a considerably difficult task.

2.6.5 Simulating quantum circuits with unbounded fan-out

The computational power of one-way quantum computation is equivalent to the power of quantum circuits with *unbounded fan-out* [25]. This model of computation consists of a generalization of the circuit model, where any two commuting gates can be performed simultaneously, thus reducing the total number of computational time slices required to implement a given computation. First explored by Høyer and Špalek [62], the unbounded fan-out model is surprisingly powerful: the quantum part of Shor's algorithm [105], for instance, can be done in one single computational step in this model.

The equivalence between the computational power of 1WQC and the unbounded fan-out model is, however, up to classical parity computations, necessary to calculate the dependency between measurement outcomes and future measurement bases in the 1WQC model. The extra power of the one-way model over the circuit model comes from its classical/quantum hybrid nature; this separation can be exploited to decrease the quantum requirements of a given algorithm by increasing its classical processing. Although this is not always possible, in Section 3.3 we explore some 1WQC optimizations techniques which use the aforementioned classical/quantum separation to reduce the number of computational steps of a given algorithm. Moreover, in Chapter 6, I show what happens when a certain family of optimized one-way quantum computations are translated back to the quantum circuit model, in an attempt to optimize quantum circuits by back-and-forth translation to the 1WQC model.

2.7 Clifford computations

In the 1WQC model, all measurements commute as quantum operations, since only single-qubit measurements are allowed and qubits are discarded after being measured. Therefore, if there is no need for adaptive choices of bases, all measurements can be performed at once in the beginning of the computation. As a matter of fact, there is a group of gates that can indeed be implemented independently of any other measurement outcome in an one-way quantum computation - the so-called *Clifford gates*. The Clifford group is generated by the CZ, H and $R_z(\pi/2)$ gates, which can all be implemented in the 1WQC model using Pauli measurements alone [more specifically, $\theta = 0$ and $\theta = \pi/2$ in Eq. (2.17)]. Therefore, any Clifford group operation can be implemented using Pauli measurements.

2.7.1 Pauli measurements and the Clifford group

The Pauli group $\mathcal{P}_n$ on n qubits is defined as the the group generated by the multiplication of n-fold tensor products of $\pm I$, $\pm iI$, X and Z. For instance, the following operators are elements of the group $\mathcal{P}_3$: $-iX \otimes i \otimes Y$, $I \otimes -iY \otimes Z$, etc. A Clifford operation C on n qubits is an operation such that for any $P \in \mathcal{P}_n$ there exists a $P' \in \mathcal{P}_n$ such that:

$$PC = CP' \tag{2.67}$$

Hence, a Clifford operator can "pass through" a Pauli operator unaltered - just the Pauli operator changes - and therefore no adaptive measurement is necessary to apply such operations in the 1WQC model. In Sec. 2.6.1, I showed how adaptive measurements can be used to counteract the by-product operator X after each measurement. However, due to the Clifford gates property captured by Eq. (2.67), the by-product Pauli operators can be handled without making use of adaptive measurements. More specifically, suppose an equation similar to Eq. (2.53) but with Clifford operators instead of J gates:

$$X^{s_4}C_4X^{s_3}C_3X^{s_2}C_2X^{s_1}C_1|\psi_0\rangle \tag{2.68}$$

Using Eq. (2.67) the equation above can be rewritten as:

$$P^sC_4C_3C_2C_1|\psi_0\rangle \tag{2.69}$$

where P is a Pauli operator and $s = f(s_1, s_2, s_3, s_4)$. The difference between Eqs. (2.69) and (2.55) is the fact that the desired unitary is implemented deterministically in Eq. (2.69), with no need for adaptive measurements. This nice property allows all measurements implementing a Clifford operator to be performed *at once* at the beginning of the computation. Even if a Clifford operation appears in the middle of a quantum circuit, the measurements that simulate those operations in the corresponding 1WQC protocol can be done at the beginning of the computation.

With such a nice property, the question of how frequently Clifford operations appear in useful QIP protocols naturally arises. As a matter of fact, Clifford gates appear in virtually every quantum algorithm or protocol, such as quantum error-correcting codes [3], Quantum Fourier Transform (QFT) [86] and the teleportation protocol described in Section 2.1.2. Thus, the way the 1WQC model computes Clifford gates may provide insights that otherwise would be difficult to see in the quantum circuit model.

2.7.2 The Gottesman-Knill theorem

Arguably the most important result regarding Clifford computation is the Gottesman-Knill theorem:

Theorem 1 ***(Gottesman-Knill Theorem** [86, 3]). A quantum computation involving only (a) state preparation in the computational basis, (b) gates in the Clifford group, (c) measurements of observables in the Pauli bases and (d) classically controlled operations depending on previous measurement outcomes can be efficiently simulated in a classical*

computer.

The original classical algorithm to simulate such circuits requires $\mathcal{O}(n^3m)$ computational steps, where n is the number of initial qubits and m is the number of steps the quantum computation requires in the circuit model. This result was further improved by Aaronson and Gottesman in [6], reducing the number of computational steps to $\mathcal{O}(n^2)$.

At first glance, one may think that since Clifford circuits can be efficiently simulated, there is no advantage coming from the fact that all Clifford circuits can be implemented in one single step in a one-way quantum computer. That is not exactly the case. Note that the classical simulation algorithm is for circuits composed only by Clifford gates, and not for a part of a generic quantum circuit composed only by Clifford group operations. On the other hand, since Pauli measurements correspond to the implementation of gates from the Clifford group, all Clifford operations can be reduced to classical processing in the 1WQC model, regardless of in which point of the quantum computation they occur. In this sense, it can be considered a stronger result than the Gottesman-knill theorem, since it provides a reduction of the quantum processing even when the circuit is composed just partially by Clifford gates.

2.7.3 Quantum-classical tradeoff

The single-step implementation of all Clifford operations in 1WQC does not come entirely for free. Suppose we want to simulate in a one-way computer a quantum circuit composed by both Clifford and non-Clifford operations. After the round of parallel Pauli measurements implementing all Clifford gates in the circuit, the exponent of the by-product operators [P^s in Eq. (2.67)] must be calculated before continuing with the implementation of the non-Clifford part. However, to calculate the exponent s (which is a modulo 2 sum) a classical computer requires $log(k)$ computational steps, where k is the number of (Pauli) measurements outcomes over which the exponent s depends upon.

Therefore, the simulation of a quantum circuit in 1WQC might not only reduce de number of total computational steps, but also trade quantum requirements for classical ones. That is, it takes a fully quantum algorithm and separates it into a quantum and classical parts, conveniently "minimizing" the quantum part. Since quantum and classical resources are usually regarded as incomparable resources[7], this separation of resources can be regarded as one of the main interesting properties of the 1WQC model.

[7]At least for technological reasons, since classical computation can be easily implemented nowadays.

It is worth noting that, regardless of the number of computational steps required, the simulation of Clifford circuits in the 1WQC model is fundamentally different from its simulation using the Gottesman and Knill (Theorem 1) approach: while the former is simulating the quantum process itself, giving a *quantum* state as output, the latter just simulates the measurement results, giving a *purely classical* output.

3 Deterministic computation using quantum measurements

In this chapter we introduce the concept of deterministic computation in the one-way model. When a quantum algorithm is implemented, it has a finite probability of success that is inherent to its construction. In the one-way model, where the information processing is driven by measurements, a central question is to determine whether the probabilistic character of quantum measurements can be compensated, such that it does not aggregate error to the algorithm being implemented.

In fact, there are several different strategies to compensate the side-effect of measuring an entangled qubit, all based in the same principle. In Sec. 2.5.1, I showed how to do so for the simple example where two qubits are entangled by a CZ gate and then one of them is measured in the $\mathcal{B}(\theta)$ basis. In that case, it was enough to apply the Pauli operator X^s to the unmeasured qubit, where s labels the measurement outcome, to guarantee the deterministic application of the unitary $U = HP(\theta)$. In this chapter I show how to construct correction strategies for arbitrary graphs (whenever possible), in order to use the associated graph states to implement deterministic computation.

As explained in Chapter 2, in this thesis I consider only single-qubit measurements in the $\mathcal{B}(\theta)$ basis applied over graph states. As a consequence of this (imposed) limitation[1], the side computer required to calculate how to adapt the measurement angles is also limited: the only calculation that needs to be performed are modulo 2 summations. A computer able to realize such summations, the so-called *parity computer*[2], is considerably less powerful than a full-fledged classical computer. Extensions of this model were discussed in [54, 55, 10], where the need of a more powerful computer (such as a classical computer) appears in the model as a way to deal with the more elaborate resource state

[1]Note that the original proposal of the one-way model has exactly the same limitation. Therefore, even with those restrictions, universal quantum computing can be achieved.

[2]The parity computer is a device implementing circuits composed only by $CNOT$ and NOT gates. The parity computer can solve a number of problems efficiently, such as simulating Clifford circuits (Gottesman-Knill theorem, see Sec. 2.7.2), and calculating the parity of bit-strings.

and/or allowed measurements.

This chapter is divided into three parts. First, in Section 3.1, I review the so-called *measurement calculus* [35], a formal language developed to work with 1WQC algorithms. In this framework every 1WQC algorithm is represented by a sequence of commands called a *measurement pattern*, which gives a description of the graph state to be created and the program (measurement bases and temporal order of operations). The composability property of measurement patterns allows the construction of 1WQC algorithms starting directly from their description in the circuit model; using a technique known as *standardization* (Sec. 3.1.4), the measurement pattern obtained by composition of small building blocks can be rewritten into a semantically equivalent form that specifies the graph state that needs to be created to perform the computation in the 1WQC model. In Section 3.2, I explain how a computation driven by quantum measurements can be made deterministic. I explain in detail what the program (measurement pattern) is doing to keep the computation deterministic and explain the different notions of determinism that arise in the one-way model.

In the last part, Sec. 3.3, I review several correcting strategies generically known as *flows*. A flow is a set of conditions over a graph that verify whether there exists a partial time ordering to perform the measurements such that a deterministic computation can be implemented (via adaptive measurements); when it exists, the flow also tells us how to adapt the measurement bases to achieve determinism. The notion of flow is different from the one of universality, discussed in Sec. 2.4. As we have seen in Def. 1, universality is defined for a family of quantum states (for instance, the family of cluster states) while one can test the flow conditions for a single graph state to see if deterministic computation (even very simple ones) can be implemented using it as a resource state. Note also that each state in a universal family of graph states satisfies the flow conditions; if it is universal it certainly implements deterministic computation. I will give a more detailed explanation on the importance of flows for 1WQC in Sec. 3.3.1.

I review three different types of flow: (1) *Regular Flow*[3] (*regflow*, for short) [33], (2) *Generalized Flow* [24] and (3) *Maximally-delayed Generalized Flow* [81]. Those properties are not mutually exclusive and therefore a graph can have two or more types of flow. A comparison between all the aforementioned flows (together with some examples) concludes the chapter.

[3]Known in the literature simply by *Flow* [33]. To avoid confusion, in this thesis I will refer to that type of flow as *Regular Flow*, letting the term *flow* to be used more freely to denote any flow-like correcting strategy.

3.1 The measurement calculus

In this section, I review the very useful formal language known as *measurement calculus* [35], developed to study MBQC. I start with a description of the commands used in the measurement calculus (Sec. 3.1.1), followed by the definition and main properties of measurement patterns (Sec. 3.1.2). Finally, I discuss two measurement calculus techniques known as *standardization* (Secs. 3.1.3 and 3.1.4) and *signal-shifting* (Sec. 5.3.1).

In this section I am going to assume that all measurement patterns given - corresponding to algorithms in the 1WQC model - are perfectly runnable (*i.e.*, physically implementable, with no anachronical operation), so that we need not to be concerned about determinism. In the following sections I show how to develop a correction strategy that allows deterministic computation (Section 3.2) and explore some of the most well-known strategies (Section 3.3).

3.1.1 Commands

The measurement calculus language consists of four types of commands. The first is the qubit initialization command N_i that prepares qubit i in the state $|+\rangle$. The input qubits are already given as an arbitrary state. The entangling command $E_{ij} \equiv CZ_{ij}$ corresponds to applying the CZ gate between qubits i and j. The single-qubit measurement command M_i^θ corresponds to a measurement of qubit i in the basis $\mathcal{B}_i(\theta) = \{|+_\theta\rangle, |-_\theta\rangle\}$, with outcome $s_i = 0$ associated with $|+_\theta\rangle$, and outcome 1 with $|-_\theta\rangle$. The measurement outcomes s_i are sometimes referred to as *signals*. The corrections may be of two types, either Pauli X or Pauli Z, and their application may be conditioned on the measurement result of several previously performed measurements. Those dependencies are denoted by $s_i = \bigoplus_{j\in J} s_j$ (with $s_j \in \{0,1\}$), where J is the set of measurements upon which the choice of basis for the measurement on qubit i depends on[4]. Therefore, the conditioned application of a Pauli X_i or Z_i are represented as X_i^s and Z_i^s, respectively, where the application of the Pauli gate depends on whether s equals zero ($Z^0 = X^0 = I$) or one.

The table below summarizes the functionality of each command:

[4]In the one-way model it is enough to use summation modulo 2 to account for all the dependencies. There are other $MBQC$ models which require a more powerful computer to calculate the dependency net of the computation, as it is the case for the models introduced in [54] where a full-fledged classical computer may be required.

Command	Functionality	Additional information
N_i	Prepares qubit i in the $\|+\rangle$ state.	Ancila preparation; represented as (unboxed) vertices in the graph.
E_{ij}	Applies the CZ gate between qubits i and j.	Entanglement operator; represented as edges in the graph.
M_i^θ	Measures qubit i in the basis $\mathcal{B}_i(\theta) = \{\|+_\theta\rangle, \|-_\theta\rangle\}$	The measurement outcome $s_i = 0$ is associated with $\|+_\theta\rangle$, and outcome 1 with $\|-_\theta\rangle$
X^s and Z^s	Applies Pauli correction X (or equivalently Z) depending on the parity of the signal s.	The signal s can be the sum (modulo 2) of many previously obtained signals.

A characteristic of the 1WQC model is that the choice of measurement bases may depend on earlier measurement outcomes. These dependent measurements can be conveniently written as ${}_t[M_i^\theta]^s$, where

$$ {}_t[M_i^\theta]^s \equiv M_i^\theta X_i^s Z_i^t = M_i^{(-1)^s\theta + t\pi}, \tag{3.1}$$

where it is understood that the operations are performed in the order from right to left in the sequence. The t and s dependencies of the measurement M_i are called its Z and X *dependencies*, respectively. Equation (3.1) can be intuitively understood by noting that, since we are dealing only with measurements in the equator of the Bloch sphere, the action of operators X and Z have straightforward consequences: The X operator flips the computational basis (*i.e.*, $|0\rangle \to |1\rangle$ and $|1\rangle \to |0\rangle$), while Z adds a "-1" phase to the eigenvector $|1\rangle$ (*i.e.*, $|0\rangle \to |0\rangle$ and $|1\rangle \to -|1\rangle$). Thus, measuring a qubit in the $\mathcal{B}(\theta)$ basis after the application of operators X^sZ^t is the same as measuring it in the $\mathcal{B}[(-1)^s\theta + t\pi]$.

3.1.2 Measurement patterns

A *measurement pattern* $\mathcal{P}$, or simply a pattern $\mathcal{P}$, is defined as a set of measurement calculus commands acting over *open graph states*, that is, quantum states associated to the so-called *open graphs*:

Definition 4 (open graph) *An open graph is a triplet (G, I, O), where $G = (V, E)$ is a undirected graph, and $I, O \subseteq V$ are the sets of, respectively, input and output vertices.*

An open graph can be obtained from a graph $G = (V, E)$ by choosing from the set of working qubits V a subset of input qubits (I) and another subset of output qubits (O).

Examples of open graphs are shown in Fig. 10. Note that not all measurement patterns are associated to physically sound sequences of operations. For example, if a given command A_j in a pattern $\mathcal{P}$ has its application conditioned on a measurement outcome s_i, then the measurement on qubit i must happen before command A_j in $\mathcal{P}$. In this thesis we will be concerned mostly with patterns that respect a few conditions of this sort, called *runnable patterns*.

Definition 5 (runnable patterns) *A measurement pattern is runnable, that is, corresponds to a physically sound sequence of operations, if it satisfies the following requirements:*

- *No command depends on outcomes of measurements that have not yet been performed;*
- *No command acts on a qubit already measured or not yet prepared, with the obvious exception of the preparation commands;*
- *A qubit undergoes measurement (preparation) iff it is not an output (input) qubit.*

As an example, take the pattern consisting of the choices $V = \{1, 2\}, I = \{1\}, O = \{2\}$ and the sequence of commands:

$$\mathcal{J}(\theta) := X_2^{s_1} M_1^{-\theta} E_{12} N_2. \tag{3.2}$$

This sequence of operations does the following: first it initialises the output qubit 2 in state $|+\rangle$; then it applies CZ on qubits 1 and 2; followed by a measurement of input qubit 1 onto the basis $\mathcal{B}(\theta)$. If the result is $|-_{\theta_i}\rangle$, then the one-bit outcome is $s_1 = 1$ and there is a correction on the second qubit ($X_2^1 = X_2$), otherwise no correction is necessary. This algorithm, compactly represented in Eq. (3.2), is the same analyzed in Section 2.5.1 and, as we now know, it implements the unitary $U = HP(\theta) = J(\theta)$. Since I am going to use the measurement pattern in Eq. (3.2) extensively throughout this thesis, from this point on I will refer to it as the *J-gate pattern*.

Next I review the definition of the *quantum depth* of a measurement pattern:

Definition 6 ***(quantum depth [25])*** *For a given open graph $G = (V, I, O)$ and associated pattern $\mathcal{P}$, we define the quantum depth of $\mathcal{P}$, or simply depth($\mathcal{P}$), as the longest subsequence (p_x) of $\mathcal{P}$, s. t. for any x, $dom(p_x) \cap dom(p_{x+1}) \neq \emptyset$, where $dom(E_{ij}) = \{i, j\}$,*

$dom(X_i^\sigma) = dom(Z_i^\sigma) = \{i\} \cup \{j \text{ s.t. } s_j \text{ appears in } \sigma\}$, $dom(M_i^\theta) = \{i\}$, $dom({}_t[M_i^\theta]^s) = \{i\} \cup \{j \text{ s.t. } s_j \text{ appears in } t \text{ or } s\}$.

Example: the measurement pattern $\mathcal{P} = X_3^{s_1+s_2} M_2^{\theta_2} X_2^{s_1} M_1^0 E_{23} E_{12} E_{13}$ associated to the graph $G = (\{1,2,3\}, \{1\}, \{3\})$ has $depth(\mathcal{P}) = 6$, since the longest dependent subsequence of $\mathcal{P}$ is $X_3^{s_1+s_2} M_2^{\theta_2} X_2^{s_1} M_1^0 E_{12} E_{13}$.

Note that the quantum depth does not take into account the depth of the classical side-processing coming from the summation mod 2 in s and t. This is so because it is taken as assumption that classical computation is free, since it is much less demanding (in terms of physical implementation) than implementing the quantum part of the algorithm. From this part on, I will refer to the quantum depth of a pattern simply as the depth of a pattern.

Measurement patterns can be combined into larger patterns. One way of combining patterns is by composition, where the output of a pattern serves as input for the next one. A pattern $\mathcal{P}_1$ can be composed with another pattern $\mathcal{P}_2$ if $V_1 \cap V_2 = I_2' = O_1'$, where $I_2' \subseteq I_2$ and $O_1' \subseteq O_1$. This is the combination used to concatenate the MBQC simulation of logical gates.

Definition 7 *Two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ can be composed if $V_1 \cap V_2 = I_2' = O_1'$, where $I_2' \subseteq I_2$ and $O_1' \subseteq O_1$, giving as result a pattern $\mathcal{P}_{12} = \mathcal{P}_1\mathcal{P}_2$ formed by the concatenation of the commands in $\mathcal{P}_1$ and $\mathcal{P}_2$ and computational space given by:*

$$V_{12} = V_1 \cup V_2, \quad I_{12} = I_1 \cup I_2'', \quad O_{12} = O_2 \cup O_1'' \tag{3.3}$$

where $I_2'' = I_2/I_2'$ and $O_1'' = O_1/O_1'$.

Composition allows, for example, to combine single-qubit gates to perform arbitrary rotations in the $SU(2)$ (as described in Sec. 2.6.1). Alternatively, patterns can be combined via a tensoring operation. This is the case if, for two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$, $V_1 \cap V_2 = \emptyset$.

Definition 8 *Two patterns $\mathcal{P}_1$ and $\mathcal{P}_2$ can be tensored if $V_1 \cap V_2 = \emptyset$, giving as result a pattern $\mathcal{P}_{12} = \mathcal{P}_1 \otimes \mathcal{P}_2$ formed by the concatenation of the commands in $\mathcal{P}_1$ and $\mathcal{P}_2$ and computational space given by:*

$$V_{12} = V_1 \cup V_2, \quad I_{12} = I_1 \cup I_2, \quad O_{12} = O_1 \cup O_2. \tag{3.4}$$

The main difference between composing and tensoring two patterns is that in the latter all unions are disjoint. As a consequence, each command of one pattern commutes with

every command in the other one. Throughout this thesis we will be concerned more about the composition than the tensoring of patterns, since the implementation of complete algorithms (composed by many gates) stands as a key feature of a universal quantum computing model, such as the one-way model. For instance, the composition of arbitrary many measurement patterns implementing the CZ gate and others implementing $J(\theta)$ gates is enough to perform arbitrary quantum computation, since $\{CZ, J(\theta)\}$ is a universal gate-set for quantum computing.

3.1.3 Command identities

Here I list some identities using commands from the measurement calculus. In the next section, these identities will be used to re-arrange commands in a given measurement pattern. Let us start with the identities involving entanglement and correction commands:

$$E_{ij}Z_i^s = Z_i^s E_{ij} \tag{3.5}$$

$$E_{ij}X_i^s = X_i^s Z_j^s E_{ij} \tag{3.6}$$

In both equations there is a qubit i that will be entangled with another qubit j, via operation E_{ij}. The identities in Eqs. (3.5) and (3.6), which can be easily verified by multiplying the matrices associated to each command (see Sec. 1.3), say what happens if the qubits are entangled before the application of the Pauli correction on qubit i. While nothing changes if the correction is a Z_i command (Eq. 3.5), if command E_{ij} switch places with X_i then a new command Z_j appears in the pattern (Eq. 3.6).

The notation ${}_t[M_i^\theta]^s$ introduced in Eq. (3.1) can be extended to comprise several different correction commands:

$${}_t[M_i^\theta]^s X_i^r = {}_t[M_i^\theta]^{s+r}, \tag{3.7}$$

$${}_t[M_i^\theta]^s Z_i^r = {}_{r+t}[M_i^\theta]^s. \tag{3.8}$$

I will also consider the following simplifications in the notation:

$${}_0[M_i^\theta]^s = [M_i^\theta]^s, \quad {}_t[M_i^\theta]^0 = {}_t[M_i^\theta], \quad {}_0[M_i^\theta]^0 = M_i^\theta. \tag{3.9}$$

Regarding the measurements of Pauli observables, I will consider the following change in notation:

$$M_i^0 = M_i^x, \tag{3.10}$$

$$M_i^{\frac{\pi}{2}} = M_i^y \tag{3.11}$$

and also the identities:

$$M_i^x X^s = [M_i^x]^s = M_i^x, \tag{3.12}$$

$$M_i^y X^s = [M_i^y]^s = {}_s[M_i^y] = M_i^y Z^s. \tag{3.13}$$

The identity in Eq. (3.12) comes from the simple fact that $(-1)^s 0 = 0$, while the one in Eq. (3.13) is a consequence of $-\frac{\pi}{2} = \frac{\pi}{2} + \pi$ (modulo 2π). In both Equations I considered that the measurement angle is given by $(-1)^s\theta + t\pi$, where s and t are the X and Z dependencies, respectively (see Eq. 3.1).

3.1.4 Standardization

In this section I review a technique introduced in [35], known as *standardization*. This technique rewrites the command sequence of a measurement pattern, by pushing the correction commands X and Z to the end (left) and the E commands to the beginning (right) of the command sequence. The expressions below, obtained by prescribing a direction (denoted by $\Rightarrow$) for the application of the identities in Sec. 3.1.3, are known as *rewrite rules* and are used in the standardization procedure:

$$E_{ij} X_i^s \Rightarrow X_i^s Z_j^s E_{ij} \tag{3.14}$$

$$E_{ij} Z_i^s \Rightarrow Z_i^s E_{ij} \tag{3.15}$$

$$E_{ij} X_j^s \Rightarrow X_j^s Z_i^s E_{ij} \tag{3.16}$$

$$E_{ij} Z_j^s \Rightarrow Z_j^s E_{ij} \tag{3.17}$$

$${}_t[M_i^\theta]^s X_i^r \Rightarrow {}_t[M_i^\theta]^{s+r} \tag{3.18}$$

$${}_t[M_i^\theta]^s Z_i^r \Rightarrow {}_{r+t}[M_i^\theta]^s \tag{3.19}$$

The following free commutation rules must also be added:

$$E_{ij} A_k \Rightarrow A_k E_{ij}, \quad \text{where } A_k \text{ is not an entanglement operator} \tag{3.20}$$

$$A_k X_i \Rightarrow X_i A_k, \quad \text{where } A_k \text{ is not a correction operator} \tag{3.21}$$

$$A_k Z_i \Rightarrow Z_i A_k, \quad \text{where } A_k \text{ is not a correction operator} \tag{3.22}$$

If a pattern $\mathcal{P}'$ is obtained from another pattern $\mathcal{P}$ by the application of a single rewrite rule [listed in Eqs. (3.14) to (3.22)], we denote the procedure of doing so by $\mathcal{P} \Rightarrow \mathcal{P}'$. Moreover, we denote by $\mathcal{P} \Rightarrow^* \mathcal{P}'$ the procedure of obtaining $\mathcal{P}'$ from another pattern $\mathcal{P}$ by the application of two or more rewrite rules.

Definition 9 ***(Standardization [35])*** *A pattern $\mathcal{P}$ is called* ***standard*** *if there is no $\mathcal{P}'$ such that $\mathcal{P} \Rightarrow \mathcal{P}'$. The process of obtaining a standard pattern $\mathcal{P}$ from another pattern is called* ***standardization****.*

A standard pattern $\mathcal{P}'$ can be obtained in a finite number of steps (*i.e.*, the standardization procedure always terminates). Moreover, all standard patterns are in the so-called $NEMC$ form[5], which means that the commands in the measurement pattern occurs in the following order: preparation commands N, entanglement commands E, measurements commands M and correction commands X and/or Z.

A few comments about the standardization procedure are in order. Regarding the complexity of the procedure, the standard form of a pattern can be obtained in at most $\mathcal{O}(N^5)$ steps, where N is the total number of qubits [35]. Now regarding the importance of the $NEMC$ form. Since all ancilla preparation and entanglement operations come first in a standard pattern, it is possible to known right from the beginning exactly which graph (and hence entanglement requirements) is necessary to perform the computation. It gives therefore a straightforward way to find out a graph 'tailored' to perform the computation in an economical way, in contrast to the use of cluster states, where several qubits need to be deleted and others "wasted" just to move a logical qubit state from a given cluster qubit to another (where a given logical gate takes place). Therefore, the standardization procedure gives task-specific graphs, which are simpler than typical, universal resources such as cluster states.

To conclude this section, let us analyze a couple of examples.

Example 1: Teleportation

Let us start with an alternative way of teleporting a qubit along a 3-qubit chain. This example was discussed in Sec. 2.5.3, where explained how a logical qubit state can be transferred from one cluster qubit to another. The pattern below is the composition of

[5]Remember that in a measurement pattern the commands in the right-most of the sequence are the ones applied first. Therefore, in the $NEMC$ form the N commands are in the right-most part of the pattern, while the C commands are the ones in the left-most of part of the pattern.

two *J-gate patterns* (Eq. 3.2), with $\theta_1 = \theta_2 = 0$.

$$\mathcal{J}_{23}(0)\mathcal{J}_{12}(0) \quad = \tag{3.23}$$

$$X_3^{s_2} M_2^0 \boxed{E_{23} X_2^{s_1}} M_1^0 E_{12} \quad \Rightarrow \quad {}_{Eq.(3.14)} \tag{3.24}$$

$$X_3^{s_2} Z_3^{s_1} \boxed{M_2^0 X_2^{s_1}} M_1^0 E_{23} E_{12} \quad \Rightarrow \quad {}_{Eq.(3.12)} \tag{3.25}$$

$$X_3^{s_2} Z_3^{s_1} M_2^0 M_1^0 E_{123} \tag{3.26}$$

Example 2: Single-qubit rotation

Here I compose the measurement pattern that implements the $J(\theta)$ gate (Eq. 3.2) in order to implement a general single-qubit rotation (as in Eq. 2.52):

$$U(\alpha, \beta, \gamma) = J(0)J(\gamma)J(\beta)J(\alpha) \tag{3.27}$$

The standadization procedure goes as follows:

$$\mathcal{J}_{45}(0)\mathcal{J}_{34}(-\gamma)\mathcal{J}_{23}(-\beta)\mathcal{J}_{12}(-\alpha) \quad = \tag{3.28}$$

$$X_5^{s_4} M_4^0 E_{45} X_4^{s_3} M_3^{-\gamma} E_{34} X_3^{s_2} M_2^{-\beta} \boxed{E_{23} X_2^{s_1}} M_1^{-\alpha} E_{12} \quad \Rightarrow \quad {}_{Eq.(3.14)} \tag{3.29}$$

$$X_5^{s_4} M_4^0 E_{45} X_4^{s_3} M_3^{-\gamma} E_{34} X_3^{s_2} \boxed{M_2^{-\beta} X_2^{s_1}} Z_3^{s_1} M_1^{-\alpha} E_{123} \quad \Rightarrow \quad {}_{Eq.(3.18)} \tag{3.30}$$

$$X_5^{s_4} M_4^0 E_{45} X_4^{s_3} M_3^{-\gamma} \boxed{E_{34} X_3^{s_2} Z_3^{s_1}} [M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{123} \quad \Rightarrow \quad {}_{Eqs.(3.14) \text{ and } (3.15)} \tag{3.31}$$

$$X_5^{s_4} M_4^0 E_{45} X_4^{s_3} Z_4^{s_2} \boxed{M_3^{-\gamma} X_3^{s_2} Z_3^{s_1}} [M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{123} \quad \Rightarrow \quad {}_{Eqs.(3.18) \text{ and } (3.19)} \tag{3.32}$$

$$X_5^{s_4} M_4^0 \boxed{E_{45} X_4^{s_3} Z_4^{s_2}}_{s_1} [M_3^{-\gamma}]^{s_2} [M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{1234} \quad \Rightarrow \quad {}_{Eqs.(3.14) \text{ and } (3.15)} \tag{3.33}$$

$$X_5^{s_4} \boxed{M_4^0 X_4^{s_3} Z_4^{s_2}} Z_5^{s_3}{}_{s_1} [M_3^{-\gamma}]^{s_2} [M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {}_{Eqs.(3.18) \text{ and } (3.19)} \tag{3.34}$$

$$X_5^{s_4} Z_5^{s_3}{}_{s_2} [M_4^0]^{s_1}{}_{s_1} [M_3^{-\gamma}]^{s_2} [M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \tag{3.35}$$

3.2 Determinism in 1WQC

To understand how a deterministic computation can be performed in the one-way model, let us start by considering what happens if we are not allowed to do any corrections or adaptive measurements. Any such sequence of operations will be of the following simple form: preparation of qubits in the $|+\rangle$ state, entanglement commands $E_{i,j}$, followed by a sequence of one-qubit measurements. Since we are looking at sequences with no corrections or dependencies, these one-qubit measurements can even be performed simultaneously, resulting in a constant-time computation. On the other hand, the map implemented on the input-output qubits will be probabilistic, instead of the deterministic,

unitary map we would like it to be.

Things would be different if we could somehow choose deterministically which outcome of each measurement $M_i^{\theta_i}$ actually happens. This is, of course, contrary to quantum theory; the computational power of such post-selected quantum mechanics has been studied by Aaronson [5], and is believed to be stronger than usual quantum theory. This notwithstanding, with postselection we can implement the map that projects each qubit onto the $|+_{\theta_i}\rangle$ state associated with outcome $s_i = 0$:

$$|\psi_{out}\rangle = \prod_{i \in O^C} P_i^{|+_{\theta_i}\rangle} |\psi_{in}\rangle \tag{3.36}$$

$$P_i^{|+_{\theta_i}\rangle} \equiv |+_{\theta_i}\rangle\langle+_{\theta_i}| \tag{3.37}$$

Note that the map is from input to output qubits, and that the projectors $P_i^{|+_{\theta_i}\rangle}$ act only on a subset of the total number of qubits, projecting the remaining qubits onto the output state $|\psi_{out}\rangle$. In post-selected quantum mechanics this map could be implemented deterministically; in the usual quantum mechanics the map can still be implemented, but with vanishingly small probability, associated with the "right" measurement outcomes $|+_{\theta_i}\rangle$ of all measured qubits. The key idea behind the one-way model is to introduce corrections and adaptive measurements that enable one to implement the map in Eq. (3.36) deterministically in some cases.

As explained in Sec. 2.4.1, a state's stabilizer is an operator that leaves the state invariant. This stabilizer's property will play an important role since it will allow some manipulations in the measurement pattern capable of, in some special cases, turn post-selected measurements into deterministic, physically implementable patterns. The stabilizer property can be mathematically expressed as:

$$K_i|G\rangle = |G\rangle, \tag{3.38}$$

where i labels a vertex from graph G and $\{K_i\}$ are the (graph-dependent) stabilizers. We refer to this equation as the application of a stabilizer operator on $|G\rangle$.

To find out how to represent a post-selected measurement using measurement calculus, we will make some map-preserving changes in the command sequence implementing the unitary $J(\theta)$ (Eq. 3.2). We start by observing that in Eq. (3.2), if we remove the correction $X_2^{s_1}$ the implementation of the unitary map would become probabilistic, *i.e.*, the map would be correctly implemented only in the cases when the measurement collapses

the qubit 1 onto its positive eigenstate $|+_\theta\rangle$. However, we can still get the same unitary map implementation of Eq. (3.2) if we post-select only the cases when the measurement result was to collapse onto the "right" eigenvector $|+_\theta\rangle$. Hence, in this context, we can say that post-selecting a measurement gives the same computation output that one would get if measure and correct the output in accordance with the measurement's result, as in Eq. (3.2). With this relation established, let us manipulate the measurement pattern to get an interesting representation of post-selection measurements. First, note that the state to be measured $|G\rangle = E_{12}N_2|\psi_{in}\rangle_1$ is stabilized by the two operators $\{Z_1^1X_2^1, Z_1^0X_2^0 = I\}$, as $Z_1X_2|G\rangle = |G\rangle$. In other words, $Z_1^{s_1}X_2^{s_1}$ is a stabilizer of $|G\rangle$ independently of whether $s_1 = 0$ or 1. Then, we can perform the following map-preserving changes on Eq. (3.2):

$$X_2^{s_1}M_1^\theta|G\rangle = X_2^{s_1}M_1^\theta X_2^{s_1}Z_1^{s_1}|G\rangle = M_1^\theta Z_1^{s_1}|G\rangle \tag{3.39}$$

Since all manipulations done did not change the unitary implemented in the measurement pattern, the last term in Eq. (3.39) is still implementing the unitary $J(-\theta)$ between input qubit 1 and output qubit 2. Note that the last term in Eq. (3.39) has a physically implausible property: the correction $Z_1^{s_1}$ needs to be performed depending on the result of an yet-unperformed measurement. On the other hand, we know by the first term in Eq. (3.39) that the map implemented is $|\psi_{in}\rangle_1 \rightarrow |\psi_{out}\rangle_2 = J(-\theta)|\psi_{in}\rangle$, which is exactly the same map that would be implemented if qubit 1 were projected onto state $|+_\theta\rangle$. Because of that, the operator $M_i^{\theta_i}Z_i^{s_i}$ has been formally used in the literature [33, 24, 81, 80, 66, 36] to represent the projection of a graph state qubit onto the state $|+_{\theta_i}\rangle$. Note, however, that the equivalence of the $M_i^{\theta_i}Z_i^{s_i}$ operator with a projection operator holds only for the remaining qubits in the graph state (that is, qubits $j \in G$ s.t. $j \neq i$); since qubits are discarded after they have been measured in the one-way model, it is not necessary that this equivalence holds also for the qubit been measured. We will explore the physical meaning of the $M_i^{\theta_i}Z_i^{s_i}$ operator in Chapter 7; for now I will assume that the following identity holds:

$$P_i^{|+_{\theta_i}\rangle}|G\rangle = M_i^{\theta_i}Z_i^{s_i}|G\rangle \tag{3.40}$$

where $i \in V$. Eq. (3.40) will play a central role throughout this section. It is straightforward to extend the measurement calculus notation for a set of post-selected measurements:

$$\prod_{i\in O^C} P_i^{|+_{\theta_i}\rangle}|G\rangle = \prod_{i\in O^C} M_i^{\theta_i}Z_i^{s_i}|G\rangle. \tag{3.41}$$

Although Eq. (3.41) cannot be implemented physically, there are indeed some cases where the stabilizer operators associated with the graph $|G\rangle$ properly cancel out all anachronical commands of form $Z_i^{s_i}$. In such cases, a deterministic computation can be physically performed.

To find out if a graph has the required stabilizer operators for deterministic computation, some sets of conditions over a graph were proposed [33, 24, 80, 81]. Those sets of conditions, known generically as *flows*, are the main subject of Section 3.3.

The r.h.s. of Eq. (3.40) is clearly an unphysical procedure in the sense that it is impossible to make the correction $Z_i^{s_i}$ as it depends of the result s_i of a measurement that has not been performed yet. To deal with this problem, we will use stabilizer operators and the fact that the square of a Pauli operator gives the identity. As an example, let us consider an entangled system represented by a graph G with associated stabilizer operators K_i. In order to perform deterministic quantum computation we want the measurements to always collapse onto the same eigenstate. For an arbitrary graph state $|G\rangle$, we have:

$$\langle +_\theta|_i \, |G\rangle = M_i^\theta Z_i^{s_i}|G\rangle \tag{3.42}$$

Now, applying a stabilizer operator on a vertex linked to i by an edge ($K_j = X_j \prod_{k\sim j} Z_k$) and using Eq. (3.38), we solve the time ordering problem:

$$\begin{aligned} M_i^\theta Z_i^{s_i} K_j^{s_i}|G\rangle &= M_i^\theta Z_i^{s_i} Z_i^{s_i} X_j^{s_i} \prod_{k\sim j,\ k\neq i} Z_k^{s_i}|G\rangle & (3.43)\\ &= \prod_{k\sim j,\ k\neq i} Z_k^{s_i} X_j^{s_i} M_i^\theta|G\rangle & (3.44) \end{aligned}$$

which is a perfectly implementable procedure, i.e., a runnable pattern (Def. 5).

Since the stabilizer operators are directly related to the graph describing the state's entanglement structure, not every command sequence can become time-respecting. Indeed, given a graph state plus the prescription of what measurements we would like to implement deterministically, it is a well-defined problem to verify if the available stabilizer operators associated to the graph G can make all measurements physically feasible (as discussed in Sec. 3.3). In other words, we need to cancel out the anachronical $Z_i^{s_i}$ of Eq. (3.41) for each measurement without introducing new ones. Since each stabilizer operator is a set of Pauli operators depending on a measurement result, it will induce a partial time-ordering in the measurement sequence. A partial order reflects the fact that not every pair of elements in the set must be related. In that way, for some pairs, it may

be that neither element precedes the order in the given set. Therefore, in our context, we can say that a partial time-ordering is the discrimination of which set of measurements can be done respecting the causality, *i.e.*, allowing only measurements that depend only on the measurements already performed.

3.2.1 Types of determinism

Due to the probabilistic nature of quantum measurements, not every measurement pattern implements a *deterministic* computation – a completely positive, trace-preserving (cptp) map that sends pure states to pure states. Given a graph G with computational space (V, I, O), let $\mathcal{H}_V$, $\mathcal{H}_I$ and $\mathcal{H}_O$ be de Hibert spaces associated with the set of qubits in V, I and O, respectively. If n is the number of measured qubits (*i.e.*, $|O^C| = n$), then the run of the associated measurement pattern $\mathcal{P}$ may follow 2^n different *branches*. Each branch is associated to a binary string $\mathbf{s}$ of length n representing the collection of classical outcomes of the measurements on that branch. Moreover, we can associate to each branch a operator A_s, called *branch map* operators, representing the linear transformation form $\mathcal{H}_I$ to $\mathcal{H}_O$ on that branch. Branch map operators can be decomposed in the following way:

$$A_s = C_s \Pi_s U \tag{3.45}$$

where C_s is a unitary operator collecting all corrections on outputs (that is, a collection of Pauli operators acting over $\mathcal{H}_O$), Π_s is a projection from $\mathcal{H}_V$ to $\mathcal{H}_O$ representing the particular set of measurements performed along that branch and U_s is the unitary embedding from $\mathcal{H}_I$ to $\mathcal{H}_V$ (preparation and entangling operators). Note that U is the same for every branch, since the associated graph G is always the same. Therefore, each measurement pattern realizes a cptp-map:

$$\begin{aligned}
\Sigma_s A_s^\dagger A_s &= \Sigma_s U^\dagger \Pi_s^\dagger C_s^\dagger C_s \Pi_s U && (3.46)\\
&= \Sigma_s U^\dagger \Pi_s^\dagger \Pi_s U && (3.47)\\
&= U^\dagger \big(\Sigma_s \Pi_s^\dagger \Pi_s\big) U && (3.48)\\
&= U^\dagger U = I. && (3.49)
\end{aligned}$$

In other words, $T(\rho) = \sum_s A_s \rho A_s^\dagger$ is a completely positive, trace-preserving map (written as a Kraus decomposition).

A measurement pattern is said to be *deterministic* if it realizes a cptp-map that sends pure states to pure states. This is equivalent to saying that branch maps are proportional.

In the case where all branch maps are equal, that is, if for every two binary strings $\boldsymbol{s_i}$ and $\boldsymbol{s_j}$ (defined as above) it holds that $A_{s_i} = A_{s_j}$, then we say the pattern is *strongly deterministic*. We say a pattern is *uniformly deterministic* if it is deterministic independently of the values of its measurement angles. Moreover, if a measurement pattern is deterministic after performing each single measurement and all the corrections depending on the result of that measurement, we say that pattern is *stepwise deterministic*. For simplicity, throughout this thesis I will call patterns which are uniformly, strongly and stepwise deterministic simply as *deterministic patterns*.

3.3 Flow theorems

In this section I address the following questions: given an open graph state, can it be used as resource state for 1WQC? If so, what can be computed using the open graph state given? To answer those questions, I will review three different sets of conditions which a graph must satisfy to be used as resource state. In this thesis, those sets of conditions will be jointly regarded as *flow conditions*, or *flow theorems*.

But what does it mean to have a graph satisfying a flow condition? In Sec. 3.2, I explained how a computation can be encoded as the (unphysical) projections of all non-output qubit. This idea is encapsulated in Eq. (3.41), which I now reproduce:

$$\prod_{i \in O^C} M_i^{\theta_i} Z_i^{s_i} |G\rangle \tag{3.50}$$

A set of flow conditions verifies if the geometry of the graph is such that there exists a set of stabilizer operators able to remove all anachronical $Z_i^{s_i}$, for $i \in O^C$, turning the measurement pattern into a runnable one (Def. 5). If a graph has this property, we say there is a *correction strategy* for that graph, which means that there exists a collection of correction Pauli operators (associated to the stabilizers K of the graph) which enable the graph to be used as resource state for deterministic 1WQC.

In the rest of this section, I review three different set of conditions, namely *regular flow*, *generalized flow* and *maximally-delayed generalized flow*. The first one is a *sufficient* condition for determinism using open graph states. Moreover, it is also *unique*: a given open graph has at most one regular flow. The generalized flow, or simply *gflow*, is a *necessary* and *sufficient* condition for determinism; it is not unique and has the regular flow as a sub-case. The last one, the maximally delayed gflow, is sometimes regarded as *optimal gflow* because it is the gflow that provides a correcting strategy with the smallest

depth. Finally, in Section 3.3.6 I give some concrete examples of the different types of flows reviewed in this section.

3.3.1 Motivation

Before going through the definitions of different types of flow, I would like to explain the relevance of the flow theorems. Perhaps the most natural question would be about the importance of such sets of conditions for determinism, since we already know several universal graph states. My answer is twofold. First, flow conditions allow one to check if a given graph state can be used to implement a deterministic computation, that is, to simulate a unitary evolution. It is specially important in the current technological status where graph states with more than a dozen qubits are considerably hard to construct. In this context, the flow conditions allow one to characterize which computations can be implemented with such restricted graph states, built given a lab's technological limitations.

The second part of my answer is about the relevance of having several flow theorems. Due to general experimental limitations, it is important to get the most out of the physical system that a lab is able to produce. And here is where the study of different sets of flow conditions come into play. Each flow theorem has its own advantages and disadvantages, that must be evaluated to better fit the lab's technological limitations. For instance, regular flow has only one element in the correcting set of each qubit, which means a reduced classical side-processing during the computation[6]. On the other hand, the maximally-delayed gflow gives the best correction strategy (which in general requires a more demanding classical side-processing) with respect to the depth of the pattern. It is, therefore, the depth-optimal flow strategy. Last but not least, the fact that the gflow function is not always unique might be useful in some scenarios. One example of is a scenario where it is not desirable that a given qubit's measurement depend on the outcome of a subset of other qubits in the graph. In a case like this, the existence of a gflow that respects such a specific requirement would be crucial for the feasibility of the protocol.

3.3.2 Regular flow

The first set of conditions we analyze is called *regular flow*, or *regflow* for short. Two properties of regflow are worth noting before we go to the formal definition. The correction

[6]It may be difficult to imagine a scenario where the reduction in the classical processing would be a relevant feature. Nevertheless, it might result in less-demanding electronic apparatus in some very specific tasks.

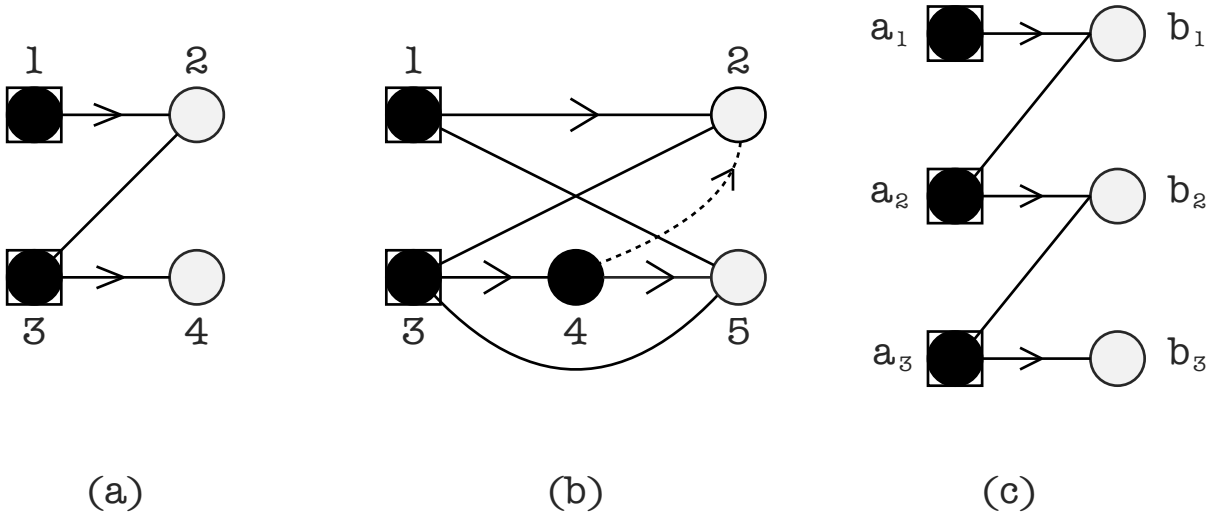


Figure 10: Examples of graphs satisfying different flow conditions. In Fig. (a) a graph with regflow is depicted, in (b) a graph with gflow but no regflow and in (c) a graph with three different gflows: a regflow, a generalized flow and an optimal gflow (see Section 3.3.6 for more information about this example). Each vertex represents a qubit, where the black vertices represent measured qubits and the white ones are unmeasured qubits. The input qubits are represented by boxed vertices and the arrows denote the dependency structure, pointing from vertex i to its correcting set. The dashed line with an arrow represent the case where the vertex from the correcting set is a non-neighbouring vertex (vertices not linked by an edge).

strategy derived from flow's set of conditions is such that, for each measured qubit $i \in O^C$, just one stabilizer operator $K_j^{s_i}$ (for some $j \in I^C$) is allowed to be used to get rid of the anachronical $Z_i^{s_i}$ in Eq. 3.41. This is clearly a limitating condition and because of that, flow is just a *sufficient* condition for deterministic computation on an open graph state.

Definition 10 (Regular flow [33]) *We say that an open graph (G, I, O) has regular flow iff there exists a map $f : O^C \to I^C$ and a strict partial order $\prec_f$ over all vertices in the graph such that for all $i \in O^C$*

- *(F1) $i \prec_f f(i)$;*
- *(F2) if $j \in N(f(i))$, then $j = i$ or $i \prec_f j$, where $N(v)$ is the neighbourhood of v;*
- *(F3) $i \in N(f(i))$;*

The set $f(i)$ is often referred to as the *regflow's correcting set* for qubit i. Efficient algorithms for finding regflow (if it exists) can be found in [36, 81]. The regflow function f is a one-to-one function. The proof is trivial, but as this property is extensively used in this work, the proof is given below:

Lemma 1 *Let $(f, \prec_f)$ be a regular flow on an open graph (G, I, O). The function f is an injective function,* i.e. *for every $i \in O^C$, $f(i)$ is unique.*

Proof. Let us assume that for some $i \in O^C$, $f(i)$ is not unique, *i.e.* there exists $j \in O^C$ such that $i \neq j$ but $f(i) = f(j)$. Then according to the flow definition:

$$j \in N(f(i)) \Rightarrow i \prec_f j,$$
$$i \in N(f(j)) \Rightarrow j \prec_f i,$$

and we arrive to a contradiction because $i \prec_f j$ and $j \prec_f i$ cannot be true at the same time. Hence $f(i)$ has to be unique. □

In the case where $|I| = |O|$, the regflow function induces a circuit-like structure in a graph, in the sense that for each input qubit $i \in I$, there exists $n \geq 0$ such that $f^n(i) \in O$, where f^n is the n^{th} iteration of the function f, and the vertex sequence

$$\{i, f(i), f^2(i), f^3(i), ..., f^n(i)\} \tag{3.51}$$

can be translated into a single wire in the circuit model. This circuit-like structure is an interesting feature of the regflow function, and it will be explored in depth in Chapters 4 and 5. An example of a graph with regflow is shown in Figure 10-a.

3.3.3 Generalized flow

Flow provides only a sufficient condition for determinism but one can generalize the above definition to obtain a condition that is both *necessary* and *sufficient*. This generalization allows correcting sets with more than one element. In those cases, we say that the graph has *generalized flow* (or simply *gflow*). In what follows let $Odd(K) = \{k\,, |N_G(k) \cap K| = 1 \mod 2\}$ to be the set of vertices where each element is connected with the set K by an odd number of edges.

Definition 11 (Generalized flow [24]) *We say (G, I, O) has generalised flow if there exists a map $g : O^C \to P^{I^C}$ (the set of all subsets of non-input qubits) and a partial order $\prec_g$ over all vertices in the graph such that for all $i \in O^C$,*

- *(G1) if $j \in g(i)$ then $i \prec_g j$;*
- *(G2) if $j \in Odd(g(i))$ then $j = i$ or $i \prec_g j$;*
- *(G3) $i \in Odd(g(i))$;*

The set $g(i)$ is often referred to as the *gflow's correcting set* for qubit i. It is important to note that flow is a special case of gflow, where $g(i)$ contains only one element. This is a key difference regarding the translation of measurement patterns to quantum circuits, as I comment in detail in Chapter 4. An example of a graph with gflow (but no regflow) is shown in Figure 10-b.

The gflow partial order leads to an arrangement of the vertices into layers, where within a given layer all the corresponding measurements can be performed simultaneously. The number of layers corresponds to the number of parallel steps in which a computation could be finished, known as the *depth* of the pattern:

Definition 12 (Depth of a gflow [81]) *For a given open graph (G, I, O) and a gflow $(g, \prec_g)$ of (G, I, O), let*

$$V_k^{\prec_g} = \begin{cases} \max_{\prec_g}(V(G)) & \text{if } k = 0 \\ \max_{\prec_g}(V(G) \setminus (\cup_{i<k} V_i^{\prec_g})) & \text{if } k > 0 \end{cases}$$

where $\max_{\prec_g}(X) = \{u \in X \text{ s.t. } \forall v \in X, \neg(u \prec_g v)\}$ is the set of maximal elements of X according to $\prec_g$. The depth *$d^{\prec_g}$ of the gflow is the smallest d such that $V_{d+1}^{\prec_g} = \emptyset$, $(V_k)_{k=0\ldots d^{\prec_g}}$ is a partition of $V(G)$ into $d^{\prec_g} + 1$ layers.*

3.3.4 Maximally-delayed Generalized flow

In [81] it was shown that a special type of gflow, called a *maximally delayed gflow*, has minimal depth.

Definition 13 (Maximally delayed gflow [81]) *For a given open graph (G, I, O) and two given gflows $(g, \prec_g)$ and $(g', \prec_{g'})$ of (G, I, O), $(g, \prec_g)$ is more delayed than $(g', \prec_{g'})$ if $\forall k$, $|\cup_{i=0\ldots k} V_i^{\prec_g}| \geq |\cup_{i=0\ldots k} V_i^{\prec_{g'}}|$ and there exists a k such that the inequality is strict. A gflow $(g, \prec_g)$ is* maximally delayed *if there exists no gflow of the same graph that is more delayed.*

In this thesis I will simply refer to the maximally delayed gflow as the *optimal gflow*, given its property of having the smallest depth possible among the gflows. Note that in [81] it was proven that the layering of the vertices imposed by an optimal gflow $(g, \prec_g)$ is always unique, whereas the gflow itself might not be unique.

3.3.5 Constructing a measurement pattern from a flow function

In the last few pages I have reviewed three different types of flow. As explained in the beginning of the Section 3.3, if a graph satisfies a flow condition it can be used for deterministic quantum computing. A natural question therefore is how one can construct a deterministic measurement pattern from a flow. It turns out that a flow function t and the associated partial order $\prec_t$ give all the information needed to build the measurement pattern associated to the open graph. A measurement pattern can be constructed in only three steps:

1. Write down Eq. (3.41) using the labels for inputs and outputs of the open graph under consideration;

2. For all $i \in O^C$, add to the measurement pattern in step 1 the set of stabilizer operators $\{K_j^{s_i}\}$ $(\forall j \in t(i))$:

$$\prod_{i \in O^C} \left[M_i^{\theta_i} Z_i^{s_i} \prod_{j \in t(i)} K_j^{s_i} \right] |G\rangle \tag{3.52}$$

3. Remove from the measurement pattern all duplicate Pauli Z and then re-organize the measurement operators M according to the partial order $\prec_t$, moving all Pauli operators depending on s_i past the M_i operator.

The final form of the measurement pattern may vary depending on the complexity of the correcting strategy. In the next section I give some examples of measurement patterns obtained from different flow functions.

3.3.6 Examples

Consider the open graph depicted in Fig. 10-c. The six-qubit graph has input set defined by $I = \{a_1, a_2, a_3\}$ and output set defined by $O = \{b_1, b_2, b_3\}$. The flow $(f, \prec_f)$ of that open graph is given by:

$$f(a_i) = b_i, \tag{3.53}$$

with partial order $a_1 \prec_f a_2 \prec_f a_3 \prec_f \{b_1, b_2, b_3\}$. The depth is therefore $d_f = 3$ (disconsidering the output layer). Following the prescription given in Section 3.3.5, the measurement

pattern associated to this flow can be constructed:

$$X_{b_3}^{s_{a_3}} M_{a_3}^{\theta_3} Z_{a_3}^{s_{a_2}} X_{b_2}^{s_{a_2}} M_{a_2}^{\theta_2} Z_{a_2}^{s_{a_1}} X_{b_1}^{s_{a_1}} M_{a_1}^{\theta_1} |G\rangle. \tag{3.54}$$

The same graph has several different gflows. For instance, a gflow that is different from both flow and optimal gflow is the following:

$$\begin{aligned} g(a_1) &= \{b_1, b_2\} \\ g(a_2) &= b_2 \\ g(a_3) &= b_3 \end{aligned}$$

with partial order $\{a_1, a_2\} \prec_g a_3 \prec_g \{b_1, b_2, b_3\}$ and depth $d_g = 2$. The associated measurement pattern is:

$$X_{b_3}^{s_{a_3}} M_{a_3}^{\theta_3} Z_{a_3}^{s_{a_2}} X_{b_2}^{s_{a_1}+s_{a_2}} M_{a_2}^{\theta_2} X_{b_1}^{s_{a_1}} M_{a_1}^{\theta_1} |G\rangle. \tag{3.55}$$

and now there is no dependency between the measurements in qubits a_1 and a_2, resulting in the depth reduction from $d_f = 3$ to $d_g = 2$.

Finally, the optimal gflow of the open graph under consideration is given by the function and partial order $(g', \prec_{g'})$:

$$\begin{aligned} g'(a_1) &= \{b_1, b_2, b_3\} \\ g'(a_2) &= \{b_2, b_3\} \\ g'(a_3) &= b_3 \end{aligned}$$

with partial order $\{a_1, a_2, a_3\} \prec_{g'} \{b_1, b_2, b_3\}$ and depth $d_{g'} = 1$. The measurement pattern is:

$$X_{b_3}^{s_{a_1}+s_{a_2}+s_{a_3}} M_{a_3}^{\theta_3} X_{b_2}^{s_{a_1}+s_{a_2}} M_{a_2}^{\theta_2} X_{b_1}^{s_{a_1}} M_{a_1}^{\theta_1} |G\rangle. \tag{3.56}$$

where no measurement depends on the result of another measurement, yielding the flat depth $d_{g'} = 1$.

4 Translating one-way patterns into the circuit model

It is well-known that the one-way model and the circuit model are equivalent, in the sense that any algorithm implemented in one model can be efficiently simulated by the other. However, the way information is processed in each model is radically different: while the later relies on unitary evolution to drive the computation, the former does so by performing measurements on a highly entangled state. With such a fundamental difference between the two models, it is reasonable (and correct) to assume that the translation of a given computation from one model to the other is not a straightforward procedure in general.

The meaning of the word *translation* can be fairly wide in some cases. In almost any case, one is specially interested in keeping the *meaning* of what is being translated. It is so because languages are used for communication purposes, and therefore it's reasonable to assume that the meaning of a given text should be the same in both source and target languages. In the context of computational models, the analogous property, the one which is desired to be invariant under translation, is the computation itself: if a computation is able to solve a set of problems in a given model, it is expected that the same set of problems can be solved when the computation is translated to a different model.

On the other hand, sometimes it is interesting to carry to the target language more than just the meaning of a text in the source language. In a poem, for instance, it is usually necessary to preserve structural properties such as metric and rhymes in order to keep its artistic value. However, if one just wants to pass on the general message of the poem, the translation of those properties would be irrelevant. Similarly, a translation procedure for computer programs may or may not preserve some of its properties. For instance, one could require that, at each computational step, both models have computed exactly the same (in opposed to just the final result being the same). Therefore, it's clear that having translation procedures able to preserve different properties can just enrich

the understanding on what are the resources needed to perform a given computation.

In this chapter I review two different translation procedures and explore their limitations and advantages. The chapter is organized as follows. In Section 4.1 I review and extend the definition of *extended circuits*, which are easily obtainable translations of measurement patterns. As we shall see, this is a straightforward method but it is inefficient in some important ways. Next, in Section 4.2, I will describe the so-called *star pattern translation* (or simply SPT). The SPT is a graphic-based translation framework that gives circuits which are less demanding (in terms of number of qubits) than the corresponding extended circuits. The SPT is, however, a very limited translation framework since it works only for measurement patterns obtained from a regular flow (Sec. 3.3.2). In Sec. 4.2.2, I show how the SPT can be used "backwards" to translate circuits to the 1WQC model, resulting in measurement patterns with regular flow. In Sec. 4.3, I explain exactly why and in which cases the SPT fails. The insights contained in that section were the starting point of the development of a new translation framework (introduced in Chapters 5 and with an application given in Chapter 6) and the study of the simulation of closed timelike curves using the 1WQC model (Chapter 7). I conclude this chapter with a brief introduction to the new translation framework, which will be introduced and explored in the next two chapters.

4.1 Extended translation

A straightforward translation method for measurement patterns, which I will refer to as *extended translation*, was introduced in [23]. This translation is inefficient, in the sense that it gives a circuit with as many wires as vertices in the original pattern (instead of having as many wires as input vertices). However its importance comes from its easy implementation, since the procedure to obtain an extended circuit is simply a re-interpretation of the measurement pattern commands using circuit notation.

Definition 14 *Given a measurement pattern with computational space (V, I, O) and underlying geometry (G, I, O) with a flow $(t, \prec_t)$. The corresponding* extended circuit C *with $|I|$ input qubits and $|V \setminus I|$ ancilla qubits, is constructed in the following steps:*

1. *Each vertex on the graph is translated as a circuit wire;*

2. *The wires corresponding to I^C are prepared in the $|+\rangle$ state;*

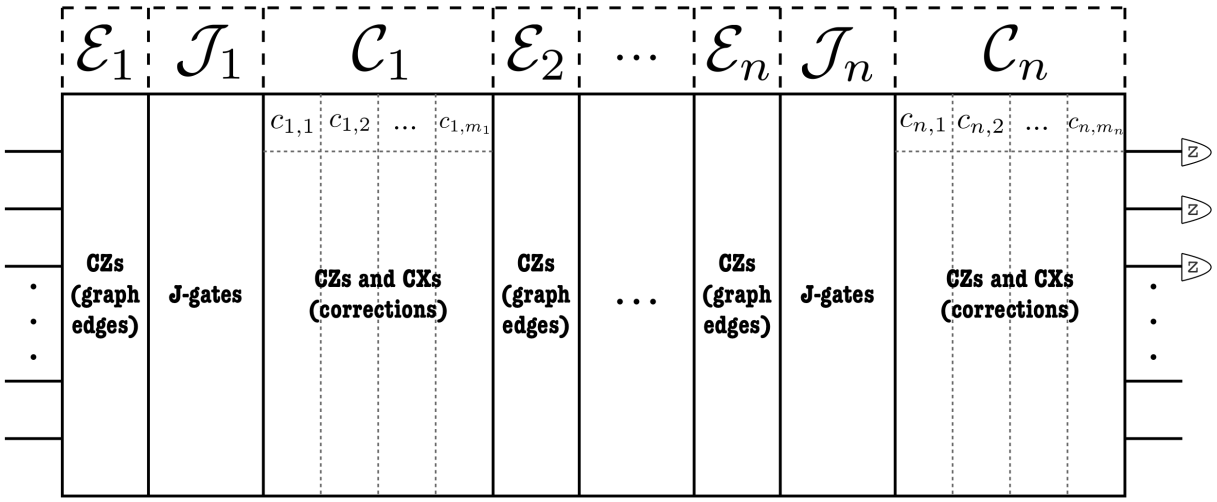


Figure 11: Generic structure of extended circuits obtained from graphs with flow or gflow. See main text for information about the division into time slices.

3. *Each edge linking vertices i and j on the graph (command E_{ij}) is translated as a CZ_{ij} gate;*

4. *Each measurement $M_i^{\theta_i}$ is translated as a gate $J(-\theta_i)$ in wire i;*

5. *Each correction command $C_j^{s_i}$ (C = Pauli -X or -Z) is translated as Controlled-C gates with qubit i as the control and j as the target wires. The layering respects $\prec_t$ by putting the control right after the J-gate and all the corresponding CX acting on qubit i;*

6. *All the qubits in O^C are measured in the computational basis in the end of the circuit.*

Let us divide the extended circuit into time slices and label these time slices as $\mathcal{E}_n$, $\mathcal{J}_n$, $\mathcal{C}_n$, which respectively include the n^{th} round of entangling, J gates and correcting gates, as illustrated in Fig. 11. We also divide slices $\mathcal{C}_n$ into s_n smaller slices $c_{n,i}$, where s_n is the total number of J-gates in slice $\mathcal{J}_n$. Each slice $c_{n,i}$ contains all correction gates with control on qubit i s.t. J_i is in $\mathcal{J}_n$.

It is easy to verify that the above circuit implements the same computation as the measurement pattern (see also [23]). For clarity, in what follows we will refer to a CZ_{ij} created in Step 3 of Def. 14 as E_{ij} while keeping the notation CZ_{ij} for those created in Step 5. Note that by construction, if a pattern is in standard form (Def. 9) then all gates associated to operators E_{ij} are initially in slice $\mathcal{E}_1$ (Step 3 in Def. 14), with $\mathcal{E}_2, ..., \mathcal{E}_n$ all empty.

At this point it is important to note some general properties of extended circuits. They contain gates of only three types: $J(\theta), CZ$ and CX. An extended circuit has as many wires as there are qubits in the associated graph, even though the goal is to implement a unitary on a much smaller subset of qubits. Each non-output qubit undergoes a single J-gate associated with its measurement in the 1WQC procedure (and wires representing output qubits have no J-gates). The state after each J-gate acts as control of possibly several CX and CZ gates acting on other wires at a time that is after their initial entangling gates (step 3) and before their respective J-gates (step 4).

As an example, let us analyze the following pattern, discussed earlier in Sec. 2.5.1[1]: $X_2^{s_1} M_1^{-\theta} E_{12} |\psi\rangle_1 |+\rangle_2$. In words, qubit 1 is an input in state $|\psi\rangle$, qubit 2 an ancilla in state $|+\rangle$; we entangle them with a E gate, then measure qubit 1 on basis $\{|\pm_\theta\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle \pm e^{-i\theta}|1\rangle)\}$. Then we correct the state of qubit 2 by applying a Pauli X conditionally on the measurement outcome s_1. The extended circuit of this measurement pattern is depicted in Fig. 12-a. Note the coherent Pauli correction: instead of a classically controlled X gate, we have applied a CX gate controlled by the first qubit's state. It is easy to check that the input-output map implemented by the circuits in Figures 12-a and 12-b is the same; we will call the identity between these circuits the *J-gate identity*. This identity will constitute an important building block of the translation framework developed in Chapters 5 and 6.

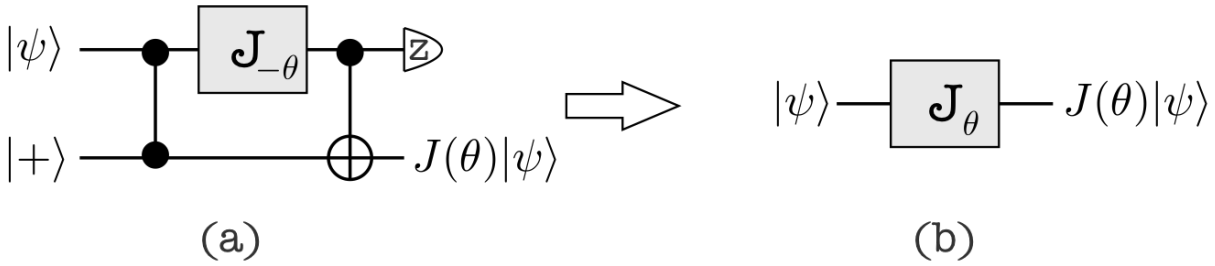


Figure 12: Extended circuit for a simple one-way quantum computation protocol. This *J-gate identity* will be used repeatedly to simplify generic extended circuits in Chapters 5 and 6. Note that the J gate angles in (a) and (b) differ by a minus sign.

The translation into an extended circuit can be obtained for any measurement pattern, as the extended circuit is just an interpretation using circuit notation of the 1WQC operations, with no attempt at optimization or adaptation. It is important to note, however, that some interesting properties of the one-way model are lost when representing it as an extended circuit. For instance, all operations in slices $\mathcal{C}$ (correction operations) of an extended circuit are performed classically on the one-way model, while in the circuit

[1] See also Eq. (3.2) and associated text.

model every operation is a coherent quantum operation.

4.2 Star pattern translation

In this section I review a more spatially economical translation procedure, the so-called *Star Pattern Translation (SPT)*. My approach will be to highlight the circuit-like structure induced by the regular flow conditions over a graph (Section 3.3.2), showing how we can use the notion of vertex sequences (Eq. 3.51) to understand how the SPT works.

4.2.1 From measurement patterns to circuits

Before reviewing the SPT itself, let us first review the concept of *vertex sequences* in graphs with regular flow, briefly mentioned in Sec. 3.3.2. Remember that the regular flow function f induces a partial ordering in the measurement pattern: for every non-output vertex i there is a vertex $f(i)$ such that $i \prec_f f(i)$ and every vertex j neighbor of $f(i)$, but different from i, obeys the relation $i \prec_f j$. Hence, the regular flow function defines 'paths' in the graph from a vertices $t_k \in I$ to vertices $s_k \in O$, such that each path can be represented as a vertex sequence like

$$\{t_k, f(t_k), f^2(t_k), ..., f^n(t_k) = s_k\} \tag{4.1}$$

For instance, consider the graph in Fig. 13-a, which has $|I| = |O| = 3$. From the regular flow definition (Def. 10), we can come up with three vertex sequences like Eq. (4.1), one for each input qubit:

$$\{t_1, f(t_1), f^2(t_1) = s_1\}, \tag{4.2}$$

$$\{t_2, f(t_2) = s_2\}, \tag{4.3}$$

$$\{t_3, f(t_3), f^2(t_3), f^3(t_3) = s_3\}, \tag{4.4}$$

These vertex sequences allow us to re-draw the graph in Fig. 13-a so as to follow the partial ordering: we arrange the vertices from the left to the right respecting the ordering of the elements in the sets in Eqs. (4.2)-(4.4) and the entanglement structure (edges), as depicted in Fig. 13-b.

We can get some intuition on how the SPT works by stating that: (1) Since i precedes $f(i)$ in the partial ordering and they are in the same vertex sequence, it must be translated

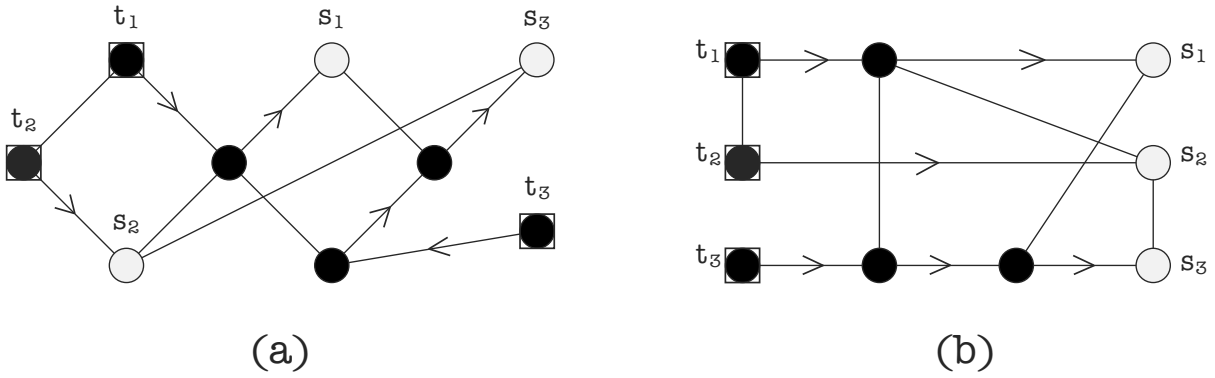


Figure 13: Two isomorphic graphs: (a) generically arranged graph and (b) same graph re-arranged to obey, from left to right, the partial ordering induced by regular flow.

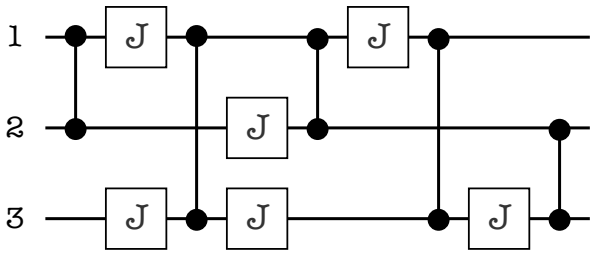


Figure 14: Quantum circuit obtained from the graph in Fig. 13-b using the Star Pattern translation.

to the circuit model as an one-qubit gate, which is how we can infer the partial ordering (the state before a one-qubit gate necessarily precedes the state after it) and (2) edges between vertices from different vertex sequences represent the two-qubit gate CZ in the circuit model. With these associations between parts of a graph and circuit elements, we obtain the quantum circuit in Fig. 14. The angle of each J-gate (omitted in the figure) is the same as the corresponding measured qubit in the measurement pattern, but with a minus sign. The circuit in Fig. 14 implements exactly the same computation as the one-way computation using the graph state in Fig. 13-a or -b.

More generally, we can decompose any graph with regular flow in several subgraphs, with one subgraph per measured qubit (Fig. 15-a). Those subgraphs can be directly translated into the circuit model, resulting in the circuit in Fig. 15-b. It is easy to verify that the one-way computation performed using the graph in Fig 15-b is the same as the one using the circuit in Fig. 15-b. These special subgraphs are associated with simple command sequences of the format:

$$X_2^{s_1} M_1^{\theta_1} E_{12} E_{13} \cdots E_{1n}, \tag{4.5}$$

where vertex 1 is a measured qubit and $2, \cdots, n$ are neighbors of qubit 1 in the graph,

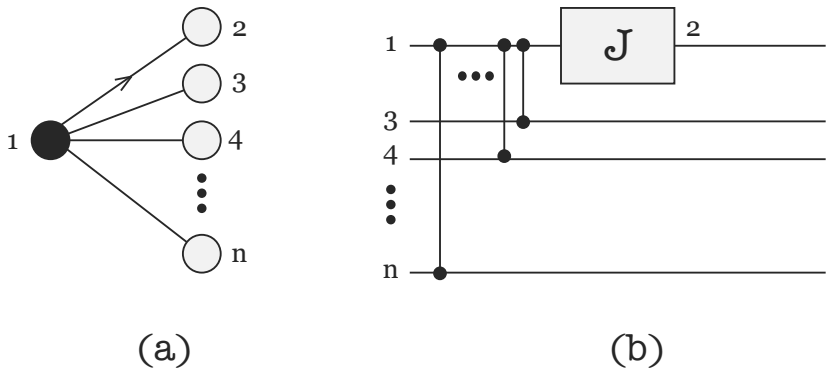


Figure 15: Fig. (a) shows a subgraph that works as a building block for the Star Pattern translation. Decomposing a general graph into these subgraphs, we can translate each one to the corresponding circuit (Fig. b) and compose them to obtain the complete translated quantum circuit.

as in Fig. 15-a. Thus, when we have a measurement pattern associated to a graph with regular flow, this means that the pattern can be fully decomposed into sub-patterns of the form of Eq. (4.5). Once we identify these pattern building blocks, the translation to the circuit model is fairly simple. The composition of the circuits originated from those subgraphs will give a quantum circuit that performs the same computation as the graph from which it was obtained. This procedure can be formally defined as follows [33]:

Definition 15 ***(Star Pattern Translation [33])*** *Let G be an open graph with computational space (V, I, O) and regular flow $(f, \prec_f)$. The corresponding quantum circuit $\mathcal{C}_G$ can be obtained by following the steps below:*

1. *Pick a vertex i in the first level of the partial order. Associate to vertex i a subgraph S_i constructed by taking vertex i as the input and all its neighbor vertices in G as output. Delete vertex i from G and repeat the process for all vertices $i \in O^C$, always respecting the partial order;*

2. *Translate each subgraph generated in Step 1 to the corresponding quantum circuit (as in Fig. 15), respecting the labels;*

3. *Combine all circuits obtained in Step 2 by connecting segments of wires with the same label.*

It is important to note, however, that the SPT cannot be used when the correcting set has more than one element, since we are not able to uniquely identify paths like Eq. (4.1) anymore. Also, from the measurement pattern framework, it means that we can not

rearrange the pattern into the simple building blocks like Eq. (4.5). Therefore, we can not apply the SPT if we want to obtain a circuit from a graph with gflow (but no regular flow). In Sec. 4.3 I show some results on the understanding of this limitation. Before analyzing the problem itself, let us see how the concept of star patterns can be used also to translate computations from the circuit model to the one-way model.

4.2.2 From circuits to measurement patterns

The decomposition of a measurement pattern into star patterns can also be used to translate quantum circuits to the one-way model. Due to the limitations on the applicability of the SPT (discussed in Sec. 4.2.1), the translation of quantum circuits using this method results always in patterns associated to graphs with regular flow. The translation from circuits to measurement patterns is implemented in a graphical level, following the steps in the definition below.

Definition 16 ***(Definition 5.2 from [23])*** *Let $\mathcal{C}$ be a circuit on n logical qubits composed by gates from the universal gateset $\{J(\theta), CZ\}$. The corresponding open graph state and associated measurement pattern $\mathcal{P}$ can be obtained by the following steps. Starting from the left of the circuit we do what follows:*

1. *Replace each J-gate by a horizontal edge connecting two vertices. Draw an arrow on the edge and label the vertex on the left as "input" and the one in the right as "output";*

2. *Replace each CZ gate by a vertical edge connecting two vertices. Label both vertices as "input/output";*

3. *The components generated during steps 1 and 2 must now be connected to form the final graph. Starting from the left, contract two non-adjacent vertices at a time according to the subcases listed below:*

   - *Two vertices labeled input/output are contracted as one vertex with input/output label;*
   - *A vertex labeled "input/output" and a vertex labeled "input" are contracted as one vertex with "input" label;*
   - *A vertex labeled "output" and a vertex labeled "input/output" are contracted as one vertex with "output" label;*

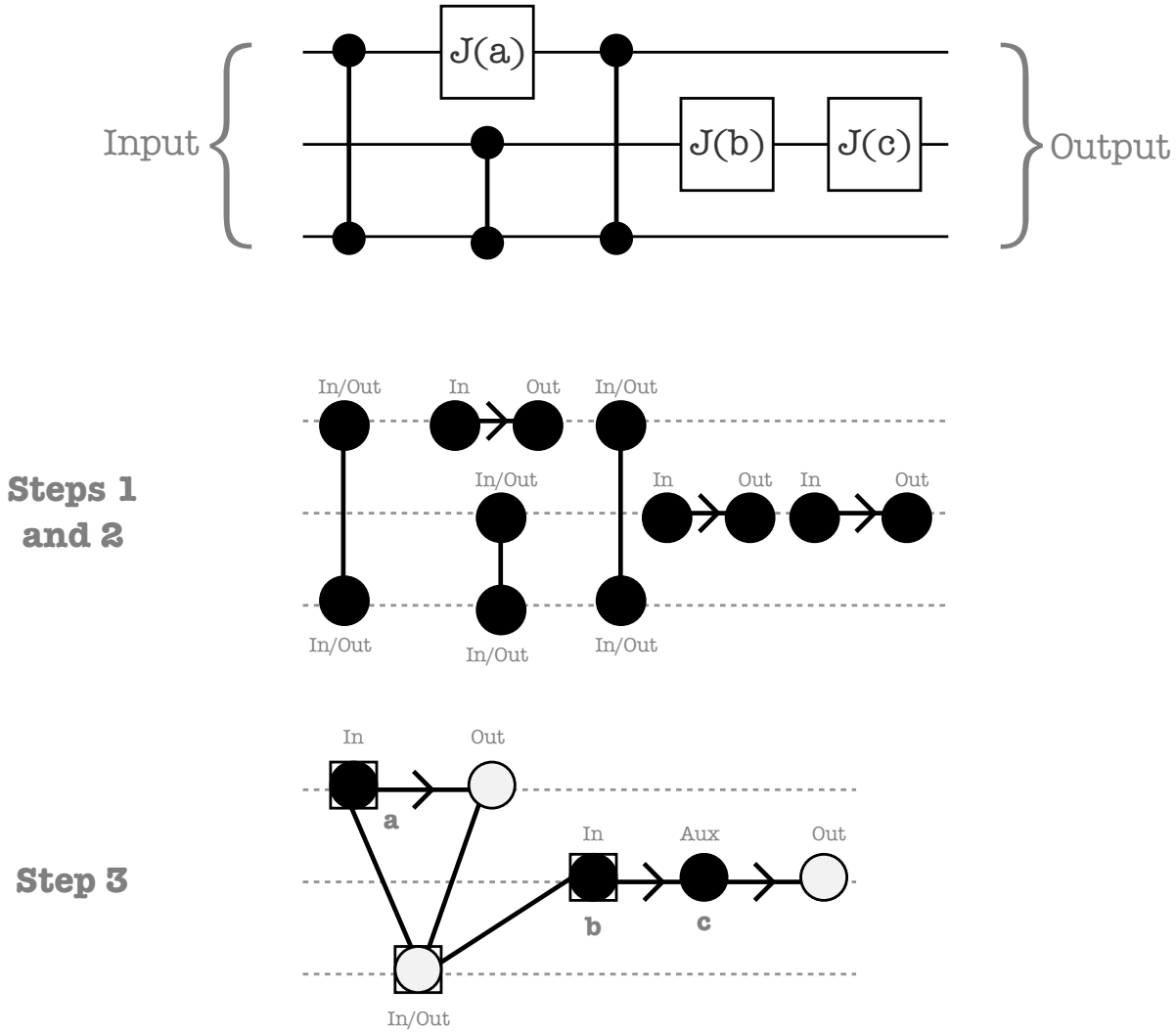


Figure 16: Using the method in Def. 16 to generate an open graph state able to simulate the circuit on the top of the figure.

- *Two vertices labeled "output" and "input" are contracted as one vertex with the label "auxiliary".*

In Fig. 16 we depict the application of the method described in Def. 16. Starting with the circuit in Fig. 16-a, Steps 1 and 2 of Def. 16 are applied, resulting in the set of subgraphs in Fig. 16-b. Then, those subgraphs are connected according to the instructions in Step 3 of Def. 16, resulting in an open graph state able to simulate the original quantum circuit. The associated measurement pattern can be inferred directly from the resulting open graph state, since both computational space (V, I, O) and flow function f are graphically specified. This method can be applied for any quantum circuit decomposed into the gateset $\{J(\theta), CZ\}$, yielding always graphs with regular flow.

4.3 The problem of using star pattern translation for patterns with no regular flow

In order to appreciate the problem that arises when applying Star Pattern Translation for gflow measurement patterns, let us apply it to the gflow pattern associated to the graph in Fig. 17-a and see what happens. First, remember that this translation method consists in decomposing a general graph into building blocks like Fig. 15-a and then substituting it by the equivalent circuit (Fig. 15-b). The composition of these sub-circuits gives the complete translated circuit. The application of this method for our example is shown in Fig. 17. There I have decomposed the complete graph into Star Pattern building blocks, resulting in the subgraphs shown in Figs. 17-b and 17-c. These subgraphs are equivalent to the circuits in Figs. 17-d and 17-e, respectively, that can be composed together (respecting the labeling) into the circuit in Fig. 17-f.

As can be noted, the translation of a perfectly runnable measurement pattern resulted in an anachronical circuit, which puts in question the validity of SPT for gflow patterns. This unexpected result for gflow pattern has been pointed out in the papers [24] and [66], where the authors claimed that it may suggest an extraordinary efficiency of gflow patterns, since the circuit originated from SPT has anachronical gates in order to preserve the same computational map of the associated measurement pattern. In this section I will show that the SPT simply does not account for every correction in the gflow's correcting sets and therefore it does not constitute a complete translation method for gflow patterns. As shown in Section 3.3.3, gflow patterns have a more complicated dependency structure, since they allow for several stabilizer operators in the measurement's correcting set (differently from the regular flow, which has just one). The SPT is ineffective for gflow patterns due to the impossibility to define a circuit-like structure in the graph (which is naturally achieved by regular flow function), always resulting in compact circuits with anachronical gates.

In order to get some insight on the origin of this problem, let us decompose the gflow measurement pattern associated to the graph in Fig. 10-b into star patterns (like Eq. 4.5). The correcting sets are $g(1) = \{3\}$ and $g(2) = \{4, 5\}$ and therefore the measurement pattern can be written as

$$X_5^{s_2} X_4^{s_2} X_3^{s_1} M_2^{\theta_2} Z_2^{s_1} M_1^{\theta_1} E_{23} E_{24} E_{14} E_{15} E_{13}. \tag{4.6}$$

We can conveniently rewrite this measurement pattern using the rules from Eqs. (3.5)

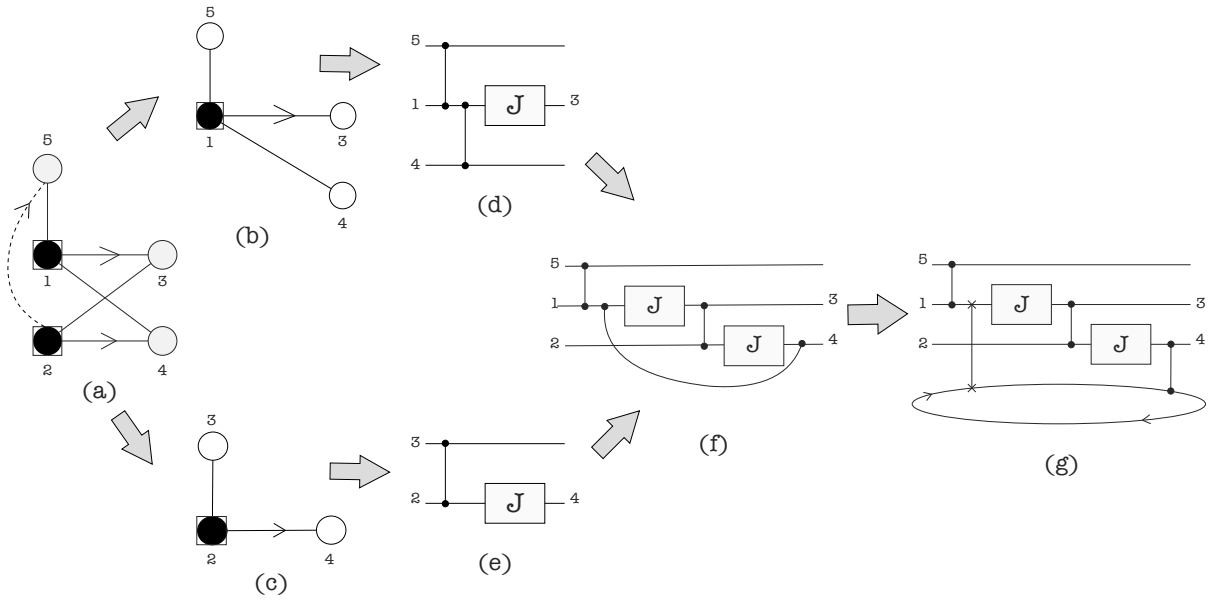


Figure 17: Graphical representation of the star pattern translation being applied to the gflow pattern associated to the graph in (a). Figures (b) and (c) show the star pattern decomposition for the graph in (a). These subgraphs can be directly translated to the circuit model using the correspondence shown in Fig. 15. As a result we have the circuits in (d) and (e). Finally, composing these subcircuits we obtain the quantum circuit in (f). We can rewrite this circuit, preserving its meaning, to the circuit in (g) by adding a $SWAP$ gate acting on the (newly added) time-traveling qubit [the round wire at the bottom of Fig. (g)] and the left-most wire segment where the anachornical CZ in Fig. (f) were acting upon. By doing so, the quantum state just before the $SWAP$ gate is able to go to the future, be acted upon by the CZ gate and then go back to the past and re-enter the time respecting part of the circuit using the same $SWAP$ gate. The circuit in (g) has no anachronical CZ like the circuit in (f) but instead there is a quantum wire interacting with a closed wire (which represents a qubit traveling in time).

and (3.6) in order to highlight the star patterns (Eq. 4.5):

$$\underbrace{X_5^{s_2}}_{Lost\ information}\ \underbrace{X_4^{s_2}M_2^{\theta_2}E_{23}E_{24}}_{Star\ pattern\ 2}\underbrace{X_3^{s_1}M_1^{\theta_1}E_{14}E_{15}E_{12}}_{Star\ pattern\ 1} \tag{4.7}$$

As can be noted in Eq. (4.7), the star pattern decomposition will only account for one stabilizer operator per measurement, which in this case are the ones associated to the vertices $s(1) = \{3\}$ and $s(2) = \{4\}$. The information that would be lost in translation is the stabilizer operator $K_5^{s_2} = X_5^{s_2}Z_1^{s_2}$. However, in Eq. (4.7) only the correction $X_5^{s_2}$ is seen because $Z_1^{s_2}$ has been canceled out with the $Z_1^{s_2}$ from the stabilizer $K_4^{s_2}$, since $Z_1^{s_2}Z_1^{s_2} = 1$.

Due to the construction of SPT, the lost information will always consist of stabilizer operators $\{K_i^{s_j}\}$ which, as we know, do not change the map implemented by the compu-

tation, since $K_i^{s_j}|G\rangle = |G\rangle$ for $i \in I^C$. However, some of those stabilizer operators are needed to eliminate anachronical Pauli corrections. Consider, for example, the graph in Fig. 10-b. As we have already seen, the SPT is not able to translate the corrections due to the stabilizer operator $K_5^{s_2} = X_5^{s_2} Z_1^{s_2}$, which means that it translates, in fact, the following measurement pattern:

$$X_4^{s_2} X_3^{s_1} M_2^{\theta_2} Z_2^{s_1} M_1^{\theta_1} Z_1^{s_2} E_{23} E_{24} E_{14} E_{15} E_{12} \tag{4.8}$$

where the last equation is obtained by removing the stabilizer operator $K_5^{s_2}$ from the measurement pattern in Eq. (4.7) (or re-applying it, since $K_5^{s_2} K_5^{s_2} = 1$). Note that now the measurement pattern has the Pauli operator $Z_1^{s_2}$, which is an operation on qubit 1 that depends on the result of a measurement (on qubit 2) not yet performed. In other words, the measurement pattern has become anachronical, not physically implementable. Remember that the equivalent quantum circuit, obtained applying the SPT directly to the graph in Fig. 17-a is shown in Fig. 17-f. Note that the anachronical Pauli operator $Z_1^{s_2}$ of Eq. (4.8) appears as a controlled-Z acting at different times in Fig. 17-f.

It is interesting to note that the anachronical circuit in Fig. 17-f can be manipulated to give an equivalent circuit without anachronical gates but that interacts with a qubit traveling in time, as seen in Fig. 17-g. This representation stands for an ordinary quantum evolution in the presence of a closed time-like curve (CTC), which is a possible solution to Einstein's General Relativity equations. Although the quantum circuit in Fig. 17-g cannot be implemented in a lab, the relation between measurement-based quantum computation and quantum circuits with time-traveling qubits can give interesting insights. This topic will be extensively discussed in Chapter 7. There I show how closed time-like curves appear naturally in the 1WQC context, studying some CTC circuits using 1WQC techniques and then comparing the results with the two main CTC models: Deutsch's [39] and the Bennett-Schumacher-Svetlichny (BSS) model [1, 106]. The one-way model encompasses the latter, having a general disagreement with the well-known Deutsch model. Furthermore, a characterization of a class of CTC circuits that admit deterministic simulation is given using stabilizer techniques associated with the one-way model.

4.4 Alternative approaches to the translation problem

Since the star pattern translation does not work properly for measurement patterns whose underlying graph has no regular flow, a new translation procedure is necessary for those patterns. An alternative approach to this problem was proposed recently in [43], where the use of a diagrammatic calculus allows one to rewrite any graph with gflow as a graph with regular flow, which can then be translated into the circuit model using the method described in Sec. 4.2.2.

It would be useful, however, to have a translation procedure that does not need an "encoding" into another formalism (such as the diagrammatic calculus). Such translation would make it easier to understand what is being changed during the translation procedure, since the circuit model is regarded as more intuitive. In the next chapter, I develop a new translation framework which does not use the SPT as a sub-routine nor the diagrammatic calculus, and yet gives the same results for graphs with regular flow, being also applicable to at least some graphs with gflow. In Chapter 6, I use this new translation framework to optimize quantum circuits by back-and-forth translation to the one-way model.

5 Compact circuits from one-way quantum computation

In this chapter I introduce a new translation framework for one-way quantum computation [40, 42]. In the previous chapters, I have reviewed some properties and limitations of the existing translation procedures: the star pattern translation (Sec. 4.2.1) gives time-respecting circuits only for a restricted class of 1WQC algorithms, namely the ones associated to regflow measurement patterns. The translation to extended circuits (Sec. 4.1), on the other hand, is able to translate all 1WQC algorithms but is remarkably inefficient in what concerns spatial resources. Moreover, the method developed in [43] (which relies on a diagrammatic calculus to do the translation) also works for any 1WQC algorithm, although with no concern in keeping some important algorithm properties, such as its depth.

The translation framework developed here relies on the relationship of the gates in the extended circuit's time slices $\mathcal{E}$ with the ones in the correction slices $\mathcal{C}$, as they appear via the use of the stabilizer formalism. This relationship is used to rewrite extended circuits with the purpose of revealing the information processing that is taking place at the logical level, that is, with logical qubits. I introduce two novel algorithms able to remove $|O^C|$ many wires from the extended circuits, while preserving the number of time slices $\mathcal{J}$ and the computation being implemented. Each algorithm is designed to work for a different class of measurement pattern: The first, designed to deal with the regflow correcting structure, gives as output circuits equivalent (in terms of number of wires and computation being implemented) to the ones given by the star pattern translation. The second algorithm works for a larger class of measurement patterns, namely the ones associates to Signal-Shifted Flow (SSF)[1]. In both cases the structure within each $\mathcal{J}$ layer is intentionally kept unchanged, allowing a better appreciation of how the number of computational steps in each model compare to each other.

[1]Recently introduced by the author and two collaborators in [42].

This chapter is divided as follows. In Section 5.1 I introduce the concept of compact circuits and the method for obtaining them from extended circuits, which I call *compactification procedures*. Section 5.2 is reserved to the development of the compactification procedure for regflow extended circuits. First, in Sec. 5.2.1, I review the regflow definition and comment on some important properties of regflow extended circuits. The algorithm that implements the compactification procedure for regflow is introduced in Sec. 5.2.2; A comparison with the existing method for regflow translation, namely Star Pattern Translation (see Sec. 4.2.1), and an example of the algorithm in use are also provided in this section. In the beginning of Section 5.3, I introduce a new type of flow called Signal-shifted Flow (SSF) and explore several of its structural properties. The SSF is more general than the regflow and its limitations and advantages are discussed in the section. An algorithm to implement the compactification procedure for SSF extended circuits is given in Sec. 5.3.5. Finally, in Sec. 5.4 I comment on the difficulty of designing compactification procedures for arbitrary flows and give a couple of examples of gflow extended circuits being transformed into their compact versions.

5.1 Compactification procedures and circuit rewriting rules

The goal of a compactification procedure is to "extract" from an extended circuit the array of gates that is being applied to the logical qubits, without explicitly calculating what the computation being performed is. In this thesis, I will call this array of gates the *compact circuit.*

There is a simple pattern from which the compact circuit can be easily obtained from its extended circuit. This is done by explicitly calculating the action of every gate in the extended circuit. This pattern was analyzed twice in this thesis (Secs. 2.5.1 and 4.1) and is called the J gate identity: $X_2^{s_1} M_1^{\theta_1} E_{12} N_2$. The associated extended circuit is in Fig 18-a. As explained before, it implements the J-gate and therefore the circuit using only logical qubits is the one shown in figure 18-b.

This method of explicitly calculating the action of every gate in the extended circuit (and also the final round of measurements in the computational basis) in order to find out the associated compact circuit is way too demanding to be applied to large circuits. Moreover, once the calculation is done, one would have only the full unitary being applied to the logical qubits; the decomposition of such unitaries into a universal gate-set - in

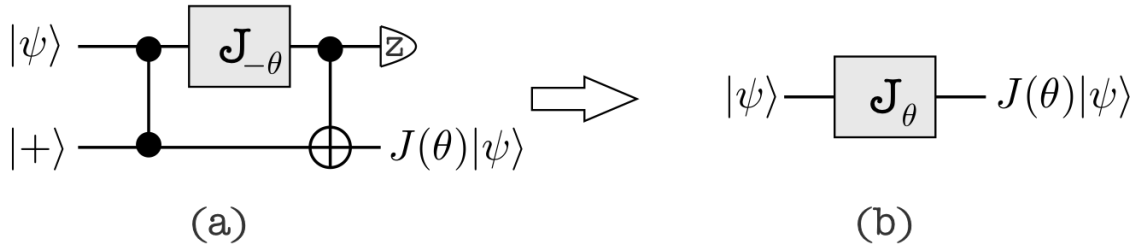


Figure 18: The *J-gate identity*. This is the same identity as the one shown in Sec. 4.1, and it is reproduced here for convenience of the reader.

order to facilitate its implementation in a physical system - is an exponentially hard task.

However, the compact circuit associated to the extended circuit in Fig. 18-a (shown in Fig. 18-b) has the remarkable property of being itself already decomposed in the universal gate-set $\{J(\theta), CZ\}$ - the same used in extended circuits. In the following sections, I develop procedures able to rewrite large extended circuits until $|O^C|$ many J-gate identities can be applied, resulting therefore in a compact circuit. This rewriting process, followed by the application of J-gate identities, is called *compactification procedure*.

A remark on notation. In this chapter, I will refer to every CZ gate the in $\mathcal{E}$ slices of an extended circuits as "E gates". The CZ gates in $\mathcal{C}$ slices will be called just "CZ gates". Moreover, every extended circuit analyzed in this chapter is assumed to have been obtained from a pattern in the standard form. As a consequence, every E gate is initially placed in the first $\mathcal{E}$ slice, that is, $\mathcal{E}_1$.

5.1.1 Compactification procedures

Compactification procedures can be described as a way of globally rewriting extended circuits in order to remove wires from the circuit without changing the computation being implemented. One way to achieve this is to rewrite the circuit to create *J-blocks*, which I now define.

Definition 17 ***(J-block)*** *Let P be a measurement pattern with underlying geometry (G, I, O) and corresponding extended circuit C. We say there is a* J-block *on wires i and j if the following set of conditions is satisfied (see Fig. 18-a):*

1. *The initial state of wire j is $|+\rangle$.*
2. *The gate sequence (E_{ij}, $J_i(\theta_i)$, CX_{ij}) appears in C.*
3. *The only gate acting on the wire j before CX_{ij} is E_{ij}.*

4. *The only gates acting on wire i after E_{ij} are $J_i(\theta_i)$ and CX_{ij} (in this order).*

5. *After the CX_{ij} gate, qubit i is measured in the Z basis.*

Once a J-block is created (via circuit rewriting), one can use the identity in Fig. 18 (J-gate identity) to remove one wire from the circuit. In general, extended circuits are not prepared for direct applications of the J-gate identity. In Definition 14, all corrections $C_j^{s_i}$ are translated as controlled-gates with control placed after the J_i gate. Hence Condition 4 in Definition 17 is not satisfied in general. Moreover, since all E gates are initially placed in slice $\mathcal{E}_1$, Condition 3 is not satisfied for general extended circuits either (see example in Figure 32-b). Thus, it is clear that in order to remove wires from extended circuits, we first need to rewrite the cicuits.

In order to create J-blocks in extended circuits, we explore the relationship between the E gates and the correcting gates C, since the latter are defined according to the former via the stabilizer formalism. In other words, there is a direct relationship between the gates in slice $\mathcal{E}_1$ and all other two-qubit gates in the rest of the extended circuit. In the case where we succeed in removing as many wires as there are non-output qubits in the graph, we say that the resulting circuit is in a *compact form*. We will refer to circuits in the compact form as *compact circuits*.

Definition 18 ***(Compactification procedure)*** *We call compactification procedure the process of rewriting extended circuits until the number of J-blocks equals the number of non-output qubits in the associated open graph, followed by the application of the J-gate identity (Fig. 18) to each J-block.*

In the next section I propose a set of circuit identities with the purpose of exploring the aforementioned relationship between two-qubit gates in extended circuits in order to create J-blocks.

5.1.2 Circuit rewrite rules

The goal of the circuit identities introduced in this section is to rewrite extended circuits, allowing the application of J-gate identities and thus the remotion of auxiliary wires. In general, each application of the J-gate identity requires a few preparatory circuit manipulations. To see why, recall that corrections associated with a given measurement M_i appear in the extended circuit as two-qubit gates (more specifically, CXs and CZs)

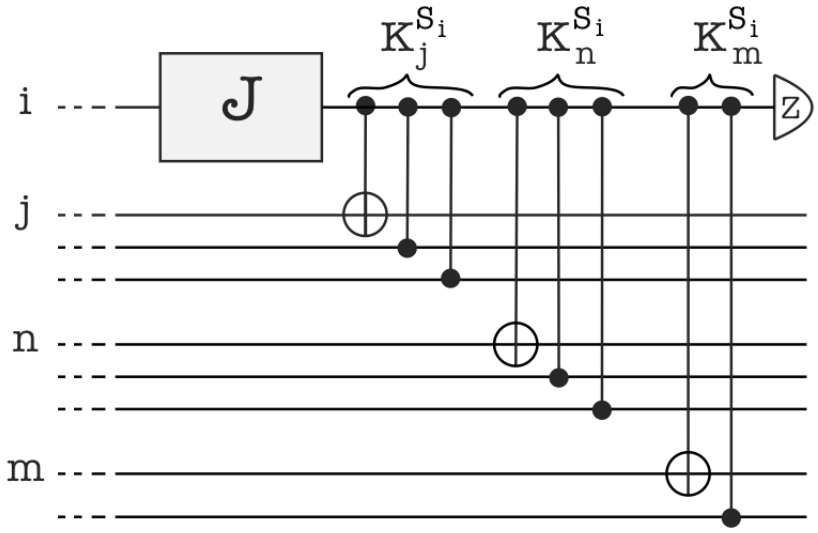


Figure 19: Extended circuit representation of a measurement and the corrections it requires on other qubits for deterministic computation in the 1WQC model. Note that each correction arises from a particular stabilizer in the correcting set of i, raised to the power s_i, where s_i is the outcome of measurement i. This sub-circuit shows only the gates corresponding to the correcting set of qubit i.

controlled by the state after the J-gate on qubit i, as in Fig. 19. While in the circuit of Fig. 18-a there is just a single CX and no CZ correction, in general extended circuits there may be other CX and CZ gates that will need to be removed if we are to use the J-gate identity to simplify the circuit.

In Fig. 20 we introduce five circuit identities that aid us in this task. The identities in Figs. 20-b and 20-c can be easily obtained from the one in Fig. 20-a, as we describe now. To start, note that the circuit identity in Fig. 20-a can be written as:

$$CZ_{ik}CX_{ij}CZ_{jk} = CZ_{jk}CX_{ij} \tag{5.1}$$

with i, j and k labeling arbitrary qubits (or vertices in a graph). Note that in this representation one must read the gates from the right-hand side to the left-hand side, while in the circuit model it is the other way around. If qubit j is in state $|+\rangle$, and since $CX_{ij}|+\rangle_j = |+\rangle_j$, we have:

$$CZ_{ik}CX_{ij}CZ_{jk}|+\rangle_j = CZ_{jk}CX_{ij}|+\rangle_j = CZ_{jk}|+\rangle_j \tag{5.2}$$

multiplying on the left by CZ_{ik}, omitting the $|+\rangle_j$ and swapping the sides (to match Figure 20-b), we have:

$$CZ_{ik}CZ_{jk} = CX_{ij}CZ_{jk} \tag{5.3}$$

which is exactly the circuit identity of of Fig. 20-b. From Eq. (5.1), considering $H_k =$

$1_i \otimes 1_j \otimes H_k$ and H_k being the Hadamard gate applied to qubit k, we also have:

$$H_k[CZ_{ik}\,(H_k CX_{ij} H_k)\,CZ_{jk}]H_k = H_k[CZ_{jk}]H_k CX_{ij} \tag{5.4}$$

Since $H_k CZ_{ik} H_k = CX_{ik}$, we get:

$$CX_{ik}CX_{ij}CX_{jk} = CX_{jk}CX_{ij} \tag{5.5}$$

which is the circuit identity in Fig. 20-c. I also consider the adjoint of the circuit identities in Fig. 20-a and 20-c, which are depicted in Fig. 20-d and 20-e, respectively.

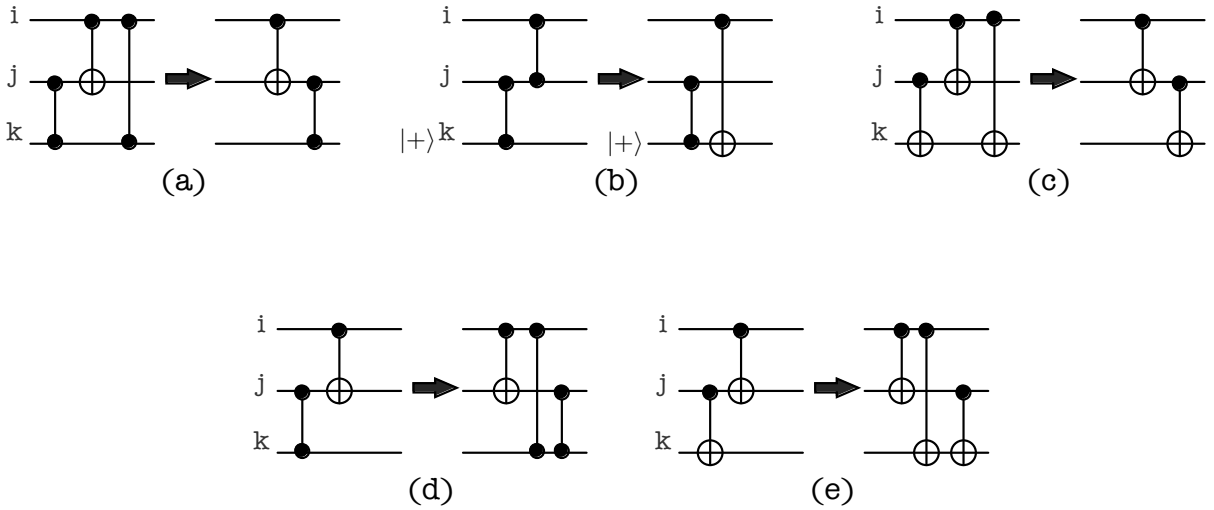


Figure 20: Five circuit identities. The circuit identity in Figure (b) is true only if qubit k is initially in the $|+\rangle$ state. The circuit identity in Figure (d) [Figure (e)] is obtained by multiplying CZ_{ik} (CX_{ik}) in both sides of the identity in Figure (a) [Figure (c)].

The circuit identities introduced in this section will be used as part of the compactification algorithms for Regular Flow (Section 5.2) and Signal-shifted Flow (Section 5.3). We will use the identities by substituting the subcircuit on the right for the one on the left in extended circuits. The two-qubit gates we need to remove from the extended circuit in order to apply J-gate identities are part of the correction structure of the original measurement pattern. Since the correction structure differs for graphs with different flows, each case must be analyzed separately.

5.2 Compact circuits from graphs with regular flow

In this section I present the compactification procedure for regflow extended circuits. First, in Sec. 5.2.1, I analyze several structural properties of the regflow function and, consequently, of regflow extended circuits, that will shed light on which rewrite rules shall

be used in order to create $|O^C|$ many J-blocks in the extended circuit. In Sec. 5.2, a full compactification algorithm for regflow extended circuit is given. Finally, a comparison with the star pattern translation (SPT, for short) is given in Sec. 5.2.2.3.

5.2.1 Regular flow structural properties

Let us start by briefly reviewing the definition of regflow. Consider a graph G for which we define a set I of input vertices and a set O of output vertices. We define a function $f : O^C \to I^C$ (from measured to prepared qubits) and a partial order $\prec_f$, where $i \prec_f j$ means that qubit j must be measured after qubit i. We say graph G has regflow if for each vertex $i \in O^C$, we can define f such that:
(F1) $i, f(i) \in G$;
(F2) $i \prec_f f(i)$;
(F3) For each k neighbor of $f(i)$ in G, with $k \neq i$, we have $i \prec_f k$.

When the entanglement graph has regflow, f is called the regflow function and identifies the stabilizer $K^{s_i}_{f(i)}$ that corrects each measurement i, and which will be translated in the extended circuit as one CX and one or more CZ gates (according to the number of neighbors of $f(i)$ in G) controlled by qubit i.

Now let us analyze how the properties of the regflow function reflect in the design of regflow extended circuits. Regflow's condition (F2) implies that, for any wire w such that there is a CX target acting upon it in slice $\mathcal{C}_i$, the gate J_w has to be in slice $\mathcal{J}_{i+1}$ or later in order to respect the partial order $\prec_f$. Equivalently, regflow's condition (F3) implies the same for CZ in a given slice $\mathcal{C}_i$. In Fig. 21-a, for instance, this means that the gates J_j and J_k must be placed in slice $\mathcal{J}_{i+1}$ or later, since there are correcting gates acting upon wires j and k in slice $\mathcal{C}_i$. Moreover, since regflow has just one element in its correcting set ($|f(i)| = 1$), each $c_{n,i}$ slice in the associated extended circuit has just one CX gate but possibly several CZ gates (In fact, $|N[f(i)]| - 1$ many[2]). In the next section I develop a compactification procedure based on these properties.

5.2.2 A compactification algorithm for regular flow extended circuits

In this section I propose an algorithm able to creat $|O^C|$ many J-blocks in a regflow extended circuit, allowing the use of J-gate identities for obtaining compact circuits. For

[2]where $N[f(i)]$ denotes the set of neighbors of vertex $f(i)$ in G.

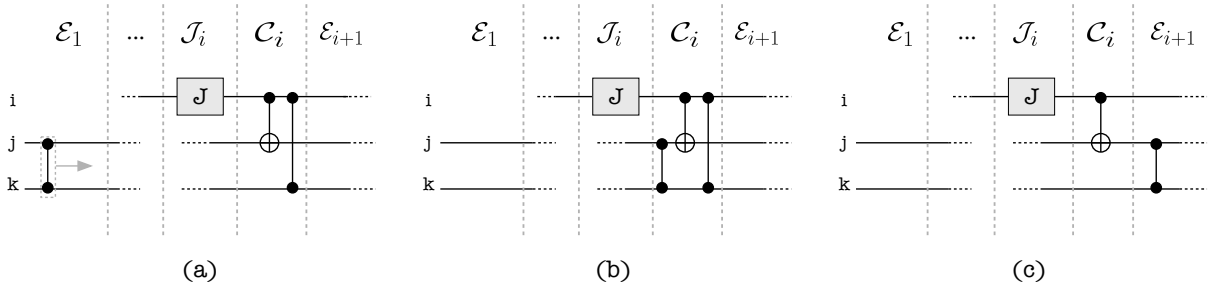


Figure 21: Rewrite Procedure 1. In Fig. 21-a we show the undesired CZ_{ik} in time slice $\mathcal{C}_i$ with the corresponding, initial entangling-round CZ_{jk} in time slice $\mathcal{E}_1$. In Fig. 21-b the CZ_{jk} gate was moved from $\mathcal{E}_1$ to $\mathcal{C}_i$ where the circuit identity in Fig. 20-a can be applied, resulting in the circuit depicted in Fig. 21-c.

graphs satisfying the regflow condition this translation into compact circuits is equivalent to the star pattern translation proposed in [23], as discussed in Sec. 5.2.2.3.

5.2.2.1 A rewrite procedure for regular flow

Let us start by recalling how the corrections in an extended circuit are associated with the graph's geometry. As we discussed in Section 3.2, to correct for the probabilistic character of the measurement on qubit i we identify operators $K_j^{s_i} = X_j^{s_i} \prod_{k\sim j} Z_k^{s_i}$, where k are the neighboring vertices of a given vertex j in the graph. In graphs with regflow, a single operator $K_j^{s_i}$ is required to correct for the measurement on i, and the vertex j associated with it is always adjacent to i. As depicted in Fig. 19, the correction operator $K_j^{s_i}$ is translated in the extended circuit as a single CX_{ij} gate together with a collection of CZ_{ik} gates, one for each $k \neq i$ adjacent to j in the graph. There is also a $Z_i^{s_i}$ operator in $K_j^{s_i}$, but this operator does not translate as a gate in the extended circuit; instead, it cancels the anachronical $Z_i^{s_i}$ operator associated with the deterministic projection (Eq. 3.41), as described previously in Sec. 3.2. As the stabilizer operators reflect the graph's geometry, for each $k \neq i$ that is adjacent to j in the graph there must be a E_{jk} gate at the beginning of the extended circuit, that is, in slice $\mathcal{E}_1$.

This means that for each CZ_{ik} gate in $\mathcal{C}$ we would like to remove (in order to create J-blocks), the extended circuit is guaranteed to have a previous E_{jk} (in slice $\mathcal{E}_1$) and also a CX_{ij} in $\mathcal{C}$ (that commutes with the CZ_{ik} gate). These three gates can be conveniently transformed using the circuit identity in Fig. 20-a, which swaps gates CX_{ij} and E_{jk}, while eliminating the troublesome CZ_{ik} gate. This procedure, of bringing the corresponding E from the beginning of the extended circuit to alongside the CZ we need to remove, followed by the application of the circuit identity in Fig. 20-a, is called Rewrite Procedure 1 (or

$RP1$ for short) and it is depicted in Fig. 21.

Now I would like to show that this removal of unwanted CZ gates can be done to any regflow extended circuits. In what follows I give a brief description of what is formally developed in Sec. 5.2.2.2 - the procedure for obtaining compact circuits from regflow extended circuits. Using the following prescription all unwanted CZs can be removed. In what follows, let W_i be the set of wires acted upon by the J-gates in slice $\mathcal{J}_i$. First, move every E gate in $\mathcal{E}_1$ not acting on a wire in W_1 to slice $\mathcal{E}_2$, commuting with the gates in $\mathcal{C}_1$. This commutation is either trivial (no gates to be commuted) or can be done using the identity in Fig. 20-a, as done in Fig. 21. With this, all unwanted CZ gates in $\mathcal{C}_1$ are removed. Now, since all E gates from the entanglement structure are in $\mathcal{E}_2$ (except for those acing on wires in W_1), the same procedure can be repeated to remove the CZ gates in $\mathcal{C}_2$, and so on until we are done. Note that, in each step, all E gates needed to remove the undesired CZs of a given slice $\mathcal{C}_i$ are placed in slice $\mathcal{E}_i$, which allows the application of the circuit identity in Fig. 20-a. Repeating this procedure for all slices, all unwanted CZ gates can be removed. In the next section I give an algorithm that implements the procedure described above[3] and show that it creates $|O^C|$ many J-gate blocks in the extended circuit.

5.2.2.2 The algorithm

Here I give an algorithm that implements the regflow compactification procedure, described in the last section. First, let me define a layering function for regflow:

Definition 19 *Let $G = (I, O, V)$ be a graph with regular flow function $(f, \prec_f)$. Define $L_f(i)$ to be a layering function from O^C into a nonzero natural number:*

$$L_f(i) = 1, \qquad if \qquad \nexists j \in V \quad s.t. \quad j \prec_f i \tag{5.6}$$

$$L_f(i) = n, \qquad if \qquad \max_{j \in V s.t. j \prec_f i} [L_f(j)] = n - 1 \tag{5.7}$$

Due to the recursive character of Def. 19, in order to evaluate $L_f(i)$ for all $i \in O^C$ one must start with vertices in the set $\{f(i) \cup N[f(i)] \setminus \{i\}\}$ for i such that $L_f(i) = 1$ (Eq. 5.6), and then moving forward doing the same for $L_f(i) = 2$, $L_f(i) = 3$ and so on. Now let us see how Algorithm 1 works.

[3]The algorithm differs from the description in Sec. 5.2.2.1 in just one point: the algorithm only moves gates from slice $\mathcal{E}_n$ to $\mathcal{E}_{n+1}$ if such procedure involves the application of a circuit identity. It is clear that doing so has no effect on the resulting compact circuit, but makes Algorithm 1 simpler.

Algorithm 1: Transforms a Regular Flow extended circuit into a compact circuit.

Input: Regular Flow extended circuit.
Output: Compact form of the inputted extended circuit.

1 **begin**
2 $D \leftarrow \max_{i \in O^C}\{L_f(i)\}$
3 **for** $n = 1$ *to* D **do**
4 **Let** $W_n = \{a_{n,1}, ..., a_{n,m_n}\}$ be the set of the m_n wires with J-gate in slice $\mathcal{J}_n$
5 **forall the** $a_{n,i} \in W_n$ **do**
6 **forall the** $k \in \{N(f(a_{n,i})) \setminus \{a_{n,i}\}\}$ **do**
7 $i \leftarrow a_{n,i}$;
8 **Apply RP1**;
9 Apply the J-gate identity for all pairs of wires $\{i, f(i)\}$ such that $i \in O^C$.

Lemma 2 *Algorithm 1 is sound and outputs compact circuits.*

Proof: The algorithm analyzes one layer of the extended circuit at a time, from $n = 1$ to $n = \max_{i \in O^C}\{L_f(i)\}$ (last layer containing non-output qubits). In each layer n, a RP1 is applied to the triad (i, j, k) where $i \in W_n$ (the set of wires with J-gate in slice $\mathcal{J}_n$), $j - f(i)$ and $k \in N(f(i)) \setminus \{i\}$. Within each layer, the application of a RP1 to wires (i, j, k) is independent from the application to another triad (i', j', k') due to the following reasons: (i) there are no two CX gates acting with target on the same wire (since f is injective) and therefore every gate in $\mathcal{C}_n$ commute to each other; (ii) Since all E gates (that will be used for the application of a RP1) commute with each other, they can be pushed to slice $\mathcal{C}_n$ in any order.

Now let us analyze what changes in each layer n after the application of all required RP1's. For every wire $i \in W_n$ the following statements hold: (i) the only gate acting on wire i after the J-gate is $CX_{if(i)}$ (all CZ gates have been removed); (ii) Every gate acting on wire $j = f(i)$ before $CX_{if(i)}$ were moved forward in the circuit with the only exception being the gate $E_{if(i)}$. In other words, after each iteration of the `for` loop the algorithm creates $|W_n|$ many J-blocks in the extended circuit. Since the `for` loop runs D times and $\sum_{n=1}^{D} |W_n| = |O^C|$, when the algorithm exits the `for` loop it has created $|O^C|$ many J-blocks. Finally, in the ninth line of the algorithm $|O^C|$ many J-gate identities are applied, giving a compact circuit as the output of the algorithm.

□

5.2.2.3 Comparison with star pattern translation

In what concerns the final compact circuit, the compactification procedure for regflow extended circuits is equivalent to the so-called *star pattern translation* [33], a graphical-based approach for translating graphs with regflow. To see this, first remember that an extended circuit has $|V|$ wires, where $|V|$ is the number of vertices in the associated graph. We have shown that for any regflow pattern our method removes $|O^C|$ wires ($|O^C|$ = number of measured qubits) from the extended circuit, resulting in a simplified circuit with $|V| - |O^C| = |O| \geq |I|$ wires. Moreover, Algorithm 1 removes all CX and CZ gates initially in slices $\mathcal{C}$, while keeping all E gates but the ones used for the application of J gate identities. Hence, the number of J gates plus the number of CZ in the final compact circuit equals the number of edges in the associated graph. The same is true for the *star pattern translation* method. The two procedures give circuits implementing the same computation with the same number of wires, and therefore both can be used to obtain compact circuits for regflow measurement patterns. One of the advantages of our method when compared to SPT is that it can be extended to some graphs that do not have regflow but which satisfy the gflow conditions, as I discuss in the next sections.

5.3 Compact circuits from graphs with signal-shifted Flow

In this section I introduce a new type of flow, which I call Signal-shifted Flow (SSF), and give a compactification procedure for extended circuits obtained from patterns with this flow. In short, a SSF pattern is a pattern obtained by applying the signal-shifting rules (Section 5.3.1) to a regular flow pattern. Therefore, only graphs with regular flow have SSF. In what follows, I briefly review the signal-shifting rules and then define the SSF (Sec. 5.3.2). Still in the same section, I explore some properties of SSF and comment on two specially interesting ones: (i) SSF is a generalized flow and (ii) if a graph has regular flow, the SSF provides the (time) optimal correcting strategy. In Section 5.3.3, I define a few structural properties of SSF that will be useful in the construction of a compactification procedure for SSF extended circuits, which is the subject of Sections 5.3.4 and 5.3.5. First, in Sec. 5.3.4, I derive a set of rewrite procedures that combine the rewrite rules introduced in 5.1.2 taking into account some relevant properties of SSF. Then, in Sec. 5.3.5, I introduce a algorithm that applies a compactification procedure to SSF extended circuits. This algorithm makes use of the rewrite procedures in Sec. 5.3.4

to completely rewrite SSF extended circuits until all non-output qubits are removed.

5.3.1 Signal-shifting

Signal-shifting is an optimization technique for measurement patterns able to reduce[4] the depth of the computation [35, 23]. The technique is basically an extension of the measurement calculus rules (Sec. 3.1) which removes from a pattern all Z-corrections acting on non-output qubits, transferring the dependency to existing X operators. In what follows, I review the signal-shifting rules and show how it works with an example. An algorithm that applies the signal-shifting rules to regflow measurement patterns is given in Sec. 5.3.1.2.

5.3.1.1 Rewrite rules

The signal-shifting rules act by changing the correction dependency of a (standard) measurement pattern. We start by substituting a S command for a Z command in the pattern using the following rule:

$$ {}_t[M_i^\alpha]^s \Rightarrow S_i^t \; [M_i^\alpha]^s \tag{5.8} $$

where the compact form of the measurement command (Eq. 3.1) was used. Then we move the S command to the left of the pattern, changing the pattern according to the following rules:

$$ {}_t[M_j^\alpha]^s \; S_i^r \Rightarrow S_i^r \; {}_{t[(r+s_i)/s_i]}[M_j^\alpha]^{s[(r+s_i)/s_i]} \tag{5.9} $$

$$ X_j^s \; S_i^t \Rightarrow S_i^t \; X_j^{s[(t+s_i)/s_i]} \tag{5.10} $$

$$ Z_j^s \; S_i^t \Rightarrow S_i^t \; Z_j^{s[(t+s_i)/s_i]} \tag{5.11} $$

where S_i^t is the signal shifting command and $s[t/s_i]$ denotes the substitution of s_i with t in s. Note that Eq. 5.9 is basically a composition of rules in Eq. 5.10 and 5.11 using the compact notation for the measurement command (Eq. 3.1).

When the S command reaches the left-most position in the pattern, it is removed. Here I assume without loss of generality that there is just one S command in the pattern at a time: a new S command is created only after the existing one has been eliminated. Moreover, since we are assuming in this thesis that all patterns are in the standard form

[4] More precisely, it is a technique that does not increase the depth of the pattern, being able to reduce it in some cases.

(Sec. 3.1.4), we do not need any commutation rule for E and S commands.

As can be seen from the rules above, signal shifting rewrites the X- and Z-corrections of a measurement pattern in a well defined manner. In particular, it will move all the Z-corrections to the end of the pattern, thereby introducing new X-corrections when Rule 5.10 is applied. It is proven in [23] that signal shifting will never increase the depth of the MBQC pattern, although it may decrease it in some cases (see below). In the case when the depth decreases, it is the consequence of the removal of the Z-corrections on the measured qubits by applying Rule 5.8.

It is important to note that the S command itself has no physical meaning, being just a formal procedure that interchanges dependencies in a measurement pattern. Therefore, a pattern containing S commands will not be physically implemented, since it may even be non-deterministic due to the incomplete correction structure of the pattern.

Example

Here I give an example of the signal shifting rules being applied to the regflow pattern. In what follows we start with the pattern in Eq. (3.35) and proceed by applying the rules being referred to after each arrow ($\Rightarrow$):

$$Z_5^{s_3} X_5^{s_4}{}_{s_2}[M_4^0]\boxed{{}_{s_1}[M_3^{-\gamma}]^{s_2}}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {\scriptstyle Eq.(5.8)} \tag{5.12}$$

$$Z_5^{s_3}\boxed{X_5^{s_4}{}_{s_2}[M_4^0]S_3^{s_1}}[M_3^{-\gamma}]^{s_2}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {\scriptstyle \text{Free commutation}} \tag{5.13}$$

$$\boxed{Z_5^{s_3} S_3^{s_1}} X_5^{s_4}{}_{s_2}[M_4^0][M_3^{-\gamma}]^{s_2}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {\scriptstyle Eq.(5.11)} \tag{5.14}$$

$$Z_5^{s_3+s_1} X_5^{s_4}\boxed{{}_{s_2}[M_4^0]}[M_3^{-\gamma}]^{s_2}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {\scriptstyle Eq.(5.8)} \tag{5.15}$$

$$Z_5^{s_3+s_1}\boxed{X_5^{s_4} S_4^{s_2}} M_4^0[M_3^{-\gamma}]^{s_2}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \quad \Rightarrow \quad {\scriptstyle Eq.(5.10)} \tag{5.16}$$

$$Z_5^{s_3+s_1} X_5^{s_4+s_2} M_4^0[M_3^{-\gamma}]^{s_2}[M_2^{-\beta}]^{s_1} M_1^{-\alpha} E_{12345} \tag{5.17}$$

Note that in Eq. 5.17 the measurement on qubit 4 does not depend on any measurement outcome. Therefore, the measurement on qubit 4 can be performed simultaneously with any other measurement in the pattern, hence reducing the depth of the computation.

5.3.1.2 Algorithm

I now present an efficient algorithm (Algorithm 2) for signal shifting a general regflow pattern (Eq. 5.18). We keep in mind that the order in which we apply the signal shifting

rules does not matter [33].

Algorithm 2: *SignalShift*

Input: A measurement pattern P with regflow $(f, \prec_f)$ as defined in Equation 5.18.
Output: The signal shifted pattern P' of P.

```
begin
    B = O^C;
    P' = P;
    while B ≠ ∅ do
        select any vertex i ∈ B which is smallest according to ≺_f;
        B = B \ {i};
        while ∃k ∈ B s.t. Z_k^{s_i} ∈ P' do
            Move the Z_k^{s_i} command next to the M_k^{α_k} command;
            Use Rule 5.8 on P' to create the signal command S_k^{s_i};
            // Removes the Z_k^{s_i} command from P';
            Use Rule 5.10 on P' to create a new X_{f(k)}^{s_i} command;
            for j ∈ N(f(k)) \ {k} do
                Use Rule 5.11 on P' to create a new Z_j^{s_i} command.
            Move S_k^{s_i} to the end of the pattern and remove it.
```

Proposition 2 ***(proposition 1 in [42])*** *Given the measurement pattern P of a regflow $(f, \prec_f)$ as defined in Equation 5.18 as input to Algorithm 2, the output will be the signal shifted measurement pattern of P.*

Proof: We will prove this proposition by showing that:

- Algorithm 2 terminates.
- Every step in Algorithm 2 that modifies the pattern P' is a valid application of a signal shifting rewrite rule.
- The output of Algorithm 2, the pattern P', is signal shifted.

We begin by showing that Algorithm 2 will terminate. The first "while" loop will obviously terminate, as we decrease the number of elements on each loop iteration and never add anything to the set B. The second "while" loop will not terminate only if some $Z_k^{s_i}$ command is added to the pattern an infinite number of times. As the underlying graph is finite and a $Z_k^{s_i}$ command represents a directed edge in the Z-correction graph, this implies the existence of a cycle in the graph, however this is impossible according to

the flow definition. The "for" loop in the algorithm terminates because the graph itself is finite, hence Algorithm 2 has to terminate.

For Algorithm 2 to actually perform the signal shifting, its operations have to be either trivial commuting rules or the three signal shifting Rules 5.8, 5.10 and 5.11. As can be easily seen from the algorithm, the operations done are indeed the signal shifting rewrite rules. We still need to prove that these rules can be applied in the order shown in the algorithm. Obviously we can use Rule 5.8 on line 8 to create the signal command due to the fact that $k \in B \subseteq O^C$ and that every non-output qubit is measured. Hence we have the measurement required for the creation of the signal command in the pattern. We know that $Z_k^{s_i}$ has to be in the pattern after the command $M_i^{\alpha_i}$ and before $M_k^{\alpha_k}$. The entanglement and creation commands are the first commands in the pattern and we do not need to move the $Z_k^{s_i}$ command past them. Hence we only need to move $Z_k^{s_i}$ past measurement commands on qubits that are not i and k and other correction commands. These can be done trivially and hence we can always move the $Z_k^{s_i}$ command next to $M_k^{\alpha_k}$ to apply Rule 5.8.

Next we want to move the newly created S_k^i command to the end of the measurement pattern. To do that we need to commute it past the commands that appear after it. The only commands S_k^i commutes non-trivially with are the ones that depend on the measurement of qubit k as can be seen from Rules 5.8 - 5.11. Those are the X- and Z-corrections depending on the measurement outcome of qubit k. According to Equation 5.18 there is exactly one such X-correction in the pattern P, namely $X_{f(k)}^k$. Also the previous steps of the algorithm could not have created any dependencies from qubit k. The Z-correction commands have only been created depending on vertices that we already moved from B. Therefore we need to create exactly one new X-correction command using Rule 5.10. We also look at the Z-corrections depending on k and from Equation 5.18 we see that in the original pattern these are on vertices from the set $N(f(k)) \setminus \{k\}$. As for the X-corrections we also have not created any new Z-corrections from k in the previous steps of the algorithm. Hence this is exactly the set of corrections we need to commute with and apply Rule 5.11. We are only left with commands after S_k^i in the pattern that commute trivially with S_k^i. We can move the command to the end of the pattern. The signal command at the end of the pattern does not influence the computation and we will not add any new commands to the end of the pattern. Hence we can remove the S_i^k command from the pattern.

Finally we show that no more signal shifting rules can be applied after the completion of Algorithm 2, *i.e.* the pattern P' is signal shifted. We eliminate all Z-corrections acting on a non-output qubit depending on a vertex i after removing it from the set B and will afterwards never create any new Z-corrections depending on that vertex. At the end of the algorithm the set B is empty, hence there cannot exist any non-output qubit that has a Z-correction acting on it and Rule 5.8 cannot be applied anymore. Moreover, since every signal command is at the end of the pattern, we cannot apply Rules 5.10 and 5.11 either. This completes the proof. □

5.3.2 The signal-shifted flow

Here I introduce and explore several properties of a new type of flow, called signal-shifted flow (SSF). The SSF measurement pattern is obtained by applying the signal shifting rules to a regflow measurement pattern. The signal shifting rules change a pattern's correction structure, which will allow us to define a flow to the associated graph. The SSF has several useful properties that we shall see in this chapter - as the existence of a SSF compactification procedure (given in Sec. 5.3.5) - and in Chapter 6 - where I use the SSF compactification procedure to construct a circuit optimization scheme.

We shall start by analyzing some properties of the signal shifting rules. The Rules 5.8 - 5.11 can be interpreted in the following way. Signal shifting takes a signal from a Z-correction on a measured qubit i (Rule 5.8) and adds it to the corrections that depend on the outcome of the measurement of i (Rules 5.9 - 5.11). If the signal was added to another Z-correction of a measured vertex, then signal shifting can be applied again until no Z-corrections are left on non-output vertices. Therefore signals move along a path created by the Z-corrections. The propagation of signals in an MBQC pattern can be described by a *Z-path* as defined below.

Definition 20 (Z-Path) *Let M be a measurement pattern on an open graph (G, I, O). Then we define a directed acyclic graph, called G_Z, on the vertices of G such that there exists a directed edge from i to j iff there exists a correction command $Z_j^{s_i}$ in M. A path in G_Z between two vertices v and u is called a* Z-path.

The above definition allows us to state a simple observation about the connectivity of a graph with flow.

Lemma 3 *If $(f, \prec_f)$ is a flow on an open graph (G, I, O), and there exists a Z-path from vertex i to vertex j, then the vertices i and $f(j)$ cannot be connected.*

Proof. The existence of a Z-path from i to j implies that $i \prec_f j$. The Z dependency graph is an acyclic graph, thus $i \neq j$. If i were connected to $f(j)$, then according to the regflow property (F2) $j \prec_f i$. Now we have two contradicting strict partial order relations $i \prec_f j$ and $j \prec_f i$. Therefore i cannot be connected to $f(j)$. □

Recall that the addition of signals is done modulo 2, therefore, if an even number of signals from a measured vertex i is added to a correction command on vertex j, the signals will cancel out (since $Z^2 = X^2 = I$). Furthermore, it is evident from the signal-shifting rewrite rules that after signal shifting, the measurement result of vertex i will create a new X-correction over vertex j if there exists an odd number of Z-paths from i to a vertex k such that j is X-dependent k in the original pattern. Similarly a new Z-correction from i to j will be created if there exists an odd number of Z-paths from i to j. Either way, *the number of Z-paths from a vertex i to another vertex j*, denoted as $\zeta_i(j)$, can be used to determine if the signal from i should be added to a correction. Also, we define $\zeta_i(i)$ to be 1 to simplify further calculations.

Before defining the *signal shifted flow* (SSF), some definitions and lemmas are needed to justify our definition. Note that if an open graph (G, I, O) has a regflow $(f, \prec_f)$, then we can write the MBQC pattern of a deterministic computation on this open graph as [33]:

$$P = \prod_{i \in O^C}^{\prec_f} \left(X_{f(i)}^{s_i} Z_{N(f(i)) \setminus \{i\}}^{s_i} M_i^{\alpha_i} \right) E_G N_{I^C} \tag{5.18}$$

where the product follows the strict partial order $\prec_f$ of the regflow $(f, \prec_f)$. From Eq. 5.18 we see that a Z-correction on a vertex j depending on the measurement outcome of another vertex i appears only if j is a neighbour of $f(i)$. This is formally stated in the next corollary as we will refer to it several times.

Corollary 1 *If (G, I, O) is an open graph with a regflow $(f, \prec_f)$, then there exists a Z-correction from vertex i to another vertex j iff $j \in N(f(i)) \setminus \{i\}$.*

We define the *Z-dependency neighbourhood* of a vertex j to be the set of vertices from which j is receiving a Z-correction. This set has an explicit form given as $N_Z(j) = \{k \in O^C | f(k) \in N(j) \setminus \{f(j)\}\}$. This is due to the following facts: first, $f(k)$ has to exist for

all vertices $k \in O^C$ because of the regflow definition; second, since $f(k) \neq f(j)$ hence k cannot be equal to j. Because $f(k) \in N(j) \Rightarrow j \in N(f(k))$ and Corollary 1 there exists a Z-correction from k to j. It is easy to see, that $\zeta_i(j)$ can be written as:

$$\zeta_i(j) = \sum_{k \in N_Z(j)} \zeta_i(k). \tag{5.19}$$

There exists a Z-correction from every $k \in N_Z(j)$ to j. These Z-corrections can be used to extend every such Z-path to k to reach j. If i is in the sum, then because $\zeta_i(i) = 1$ the correct number of Z-paths is obtained with Eq. 5.19.

As mentioned before, the evenness of the number of Z-paths can be used to determine if a signal is added to a correction command. Let $parity(n)$ be the function that determines the oddness or evenness of the integer n, *i.e.* $parity(n) = n \mod 2$. Then if an open graph has a regflow, the oddness of $\zeta_i(j)$ can be found as described in the following lemma.

Lemma 4 ***(lemma 5 in [42])*** *For every two vertices i and j in an open graph (G, I, O) with regflow $(f, \prec_f)$*

$$parity(\zeta_i(j)) = |k \in \{N_Z(j) | parity(\zeta_i(k)) = 1\}| \mod 2$$

i.e. $parity(\zeta_i(j))$ *depends only on the number of vertices in the Z-dependency neighbourhood which have odd number of Z-paths from i.*

Proof: The oddness of $\zeta_i(j)$ can be written as

$$\begin{aligned} parity(\zeta_i(j)) &= \left(\sum_{k \in N_Z(j)} \zeta_i(k) \right) \mod 2 = \\ &= \sum_{k \in N_Z(j)} (\zeta_i(k) \mod 2) \mod 2 = \\ &= \sum_{\{k \in N_Z(j) | 1 = \zeta_i(k) \mod 2\}} (\zeta_i(k) \mod 2) \mod 2 = \\ &= |\{k \in N_Z(j) | 1 = \zeta_i(k) \mod 2\}| \mod 2 = \\ &= |\{k \in N_Z(j) | parity(\zeta_i(k)) = 1\}| \mod 2 \end{aligned}$$

□

All these notions will allow us to define the structure of the pattern after signal shifting has been performed.

Proposition 3 ***(proposition 2 in [42])*** *Given a regflow $(f, \prec_f)$ on an open graph*

(G, I, O), let s be a function from $O^C \mapsto P^{I^C}$ such that $j \in s(i)$ iff $parity(\zeta_i(f^{-1}(j))) = 1$. Also define L_s to be a layering function from $V(G)$ into a natural number:

$$L_s(i) = 0 \qquad \forall i \in O$$
$$L_s(i) = \max_{j \in s(i)}(L_s(j) + 1) \qquad \forall i \notin O$$

Define the strict partial order $\prec_s$ with:

$$i \prec_s j \quad \Leftrightarrow \quad L_s(i) > L_s(j)$$

Then, the application of signal shifting Rules 5.8 - 5.11 over an MBQC pattern with regflow $(f, \prec_f)$ will lead to the following pattern:

$$P = \prod_{j \in O, i \in I^C} Z_j^{s_i . parity(\zeta_i(j))} \prod_{i \in O^C}^{\prec_s} \left(X_{s(i)}^{s_i} M_i^{\alpha_i} \right) E_G N_{I^C} \tag{5.20}$$

Proof: The proof is divided into three parts. First we will show that signal shifting creates exactly the pattern commands shown in Equation 5.20. We proceed by showing that the layering function L_s is defined for every $i \in V(G)$. Lastly, we need to prove that using the partial order $\prec_s$ derived from L_s for ordering the commands as in Equation 5.20 gives a valid measurement pattern.

Note that the preparation commands (N_I^C), entanglement commands (E_G) and measurement commands ($M_i^{\alpha_i}$) are the same for Equations 5.18 and 5.20. Because signal shifting does not change these commands (Rules 5.8 - 5.11) these are as required for a signal shifted pattern. Hence we need only to consider the correction commands.

We will look at the correction commands that would appear in a signal shifted pattern. We do this by examining the signal shifting algorithm (Algorithm 2). As mentioned before, the algorithm works as a directed graph traversal, in a way that every distinct path is traversed. As seen in the algorithm every $Z_k^{s_i}$ correction acting on a non-output qubit is removed from the pattern. This is in accordance with the proposed pattern in Equation 5.20. Let us examine which new corrections are created.

The number of newly created $X_j^{s_i}$ depends on the number of times we enter the first loop with command $Z_{f^{-1}(j)}^{s_i}$. As the algorithm is a directed graph traversal algorithm, this happens as many times as there are different paths over the Z-dependency graph from i to $f^{-1}(j)$. Because the same two $X_l^{s_i}$ corrections cancel each other, hence a new X-correction appears in a signal shifted pattern only if $parity(\zeta_i(f^{-1}(j))) = 1$. We also note that no new $X_{f(i)}^{s_i}$ correction is created since there exist no Z-path between i and $f^{-1}(f(i))$. On

the other hand Algorithm 2 leaves the already existing X corrections unchanged and moreover since we have defined $\zeta_i(i) = 1$ therefore $f(i) \in s(i)$. This implies that the set $s(i)$ does indeed contain all the vertices that have an X-correction depending on s_i after signal shifting is performed.

The number of newly created Z-corrections on an output vertex j depending on a vertex i appearing in the signal shifted pattern is equal to the number of different paths from i to j. The difference with non-output qubits is that these will not be removed through the process of signal shifting. As with X-corrections, two Z-correction commands on the same qubit will cancel each other out and hence the existence of a $Z_j^{s_i}$ in the final pattern depends on the parity of the number of paths from i to j. This can be written in short as:

$$Z_j^{s_i \cdot parity(\zeta_i(j))}$$

Hence the measurement pattern in Equation 5.20 has exactly the same commands as the signal shifted pattern in Equation 5.18.

Another thing we need to prove is that the layering function L_s is defined for every $i \in V(G)$. As proven above, the X-corrections depending on the measurement of qubit v correspond to the set $s(v)$. Hence we can interpret the definition of $L_s(v)$ as finding the maximum value of L_s for every vertex that has an X-correction from v and adding 1 to it. The recursive definition of $L_s(v)$ is well defined, if for every non-output qubit we can find a path over X-corrections ending at an output qubit. We know that signal shifting of a valid pattern creates another valid pattern. This implies that the X-corrections cannot create a cyclic dependency structure and hence every path over the X-corrections has an endpoint. Moreover such a path cannot end on a non-output qubit k since $f(k) \in s(k)$ and one could always extend that path with $f(k)$. Therefore $L_s(v)$ is well defined.

Finally, it is easy to show that the partial order $\prec_s$ as used in Equation 5.20 gives a valid ordering of the commands. Every vertex j that has an X-correction depending on the measurement of qubit i has a smaller L_s number and hence $i \prec_s j$. This way no X-correction command acts on an already measured qubit and because the Z-corrections are applied only on output qubits, the correction ordering is valid. Every other command is applied before the measurement command and hence the pattern in Equation 5.20 is a valid measurement pattern. □

Given an open graph with regflow, we refer to the construction of the above proposition as its corresponding *signal-shifted flow* (SSF).

Corollary 2 ***(corollary 2 in [42])*** *If (G, I, O) is an open graph with regflow $(f, \prec_f)$ and SSF $(s, \prec_s)$ then for every vertex i and j such that $f(j) \in s(i) \setminus \{f(i)\}$, we can find another vertex k, such that $f(k) \in s(i) \cap N(j)$.*

Proof: If $f(j) \in s(i)$, then from the Proposition 3 of SSF we can conclude that $parity(\zeta_i(j)) = 1$. We know that $j \neq i$ from the assumptions. Lemma 4 says that there must exist at least one other vertex k from which j has a Z-correction, such that $parity(\zeta_i(k)) = 1$. The regflow definition says that j must therefore be a neighbour of $f(k)$. Definition 3 of SSF states that $f(k)$ must therefore be in $s(i)$, hence $f(k) \in s(i) \cap N(j)$. □

The SSF has several interesting properties, but two of them are specially relevant for this thesis. Both properties were proved in [42] and are stated in the theorems below. The proofs were omitted since they are rather long and are out of the scope of this thesis. The two SSF properties stated in the theorems below have motivated the developement of a SSF compactification procedure (Sec. 5.3.5) and the circuit optimization scheme analyzed in Chapter 6.

Theorem 2 ***(theorem 1 from [42])*** *Given any open graph (G, I, O) with flow $(f, \prec_f)$, the corresponding signal shifted regflow $(s, \prec_s)$ is a gflow.*

Theorem 3 ***(theorem 2 from [42])*** *Let (G, I, O) be an open graph with regflow $(f, \prec_f)$ such that $|I| = |O|$. Let $(s, \prec_s)$ be the SSF obtained from $(f, \prec_f)$. Then $(s, \prec_s)$ is the optimal gflow of (G, I, O).*

Therefore, if a graph has regflow, the SSF provides the optimal correcting strategy (with respect to the depth of the pattern). Moreover, the SSF flow is easier to find than the optimal gflow introduced in [81], specially due to the fact that SSF pattern can be constructed by the application of the simple rewrite rules in Eqs. 5.8 to 5.11[5].

5.3.3 SSF structural properties

The notions of *influencing walks* and *partial influencing walks* on open graphs with flow were introduced in [23] to describe the set of all vertices that a measurement depends

[5]Note, however, that it is valid only for graphs with regflow. Since SSF is not defined for graphs that do not satisfy regflow conditions, the method of [81] is the only known method for finding the optimal gflow in those cases.

on. An influencing walk starts with an input and ends with an output vertex, a partial influencing walk starts with an input vertex but can end with a non-output vertex. We will use a modified definition of influencing walks that can start from any non-output vertex i and end at any vertex $j \in s(i)$ and call it a *stepwise influencing path*. This will allow us to conveniently explore the dependency structure of a pattern with SSF.

Definition 21 *Let $(s, \prec_s)$ be an SSF that is obtained from a regflow $(f, \prec_f)$ of an open graph (G, I, O) and vertices i and j in $V(G)$ such that $j \in s(i)$. We say that a path between vertices i and j is an **stepwise influencing path**, noted as $\wp_i(j)$, iff*

- *The path is over the edges of G.*
- *The first two elements on the path are i and $f(i)$.*
- *Every even-placed vertex k on the path $\wp_i(j)$, starting from $f(i)$, is in $s(i)$.*
- *Every odd-placed vertex on the path $\wp_i(j)$ is the unique vertex $f^{-1}(k)$ of some $k \in s(i)$ such that k is the next vertex on the path $\wp_i(j)$.*

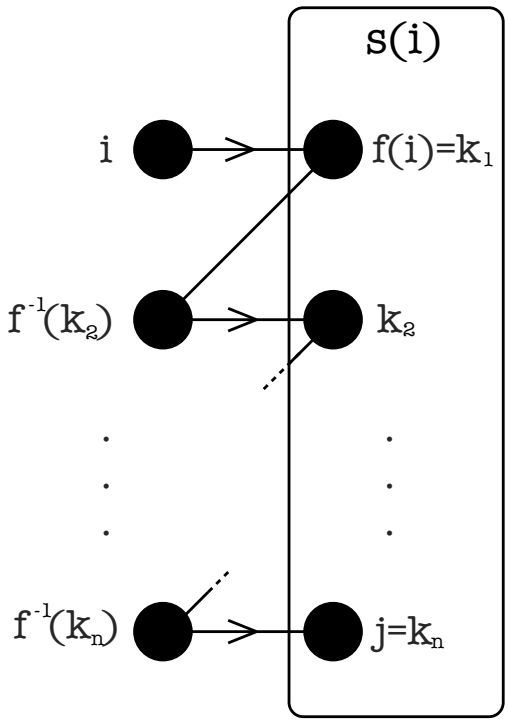


Figure 22: Stepwise influencing path $\wp_i(j)$. See Def. 21.

It is easy to see that every second edge, in particular the edges between $f^{-1}(k)$ and $k \in s(i)$, in the stepwise influencing path is a flow edge. Hence the path contains no

consecutive non-flow edges. If we restrict the first vertices of the stepwise influencing path to be input vertices, the stepwise influencing path would be a partial influencing path, but not *vice versa*. Stepwise influencing paths are useful because of their appearance in the SSF as proven by the following lemma.

Lemma 5 *Let $(s, \prec_s)$ be an SSF obtained from a regflow $(f, \prec_f)$ of an open graph (G, I, O) and vertices i and j in $V(G)$ such that $j \in s(i)$. Then there always exists a stepwise influencing path $\wp_i(j)$.*

Proof: We start by constructing such a path backward from j to i. We select j and $f^{-1}(j)$ as the last two vertices on the path and apply Corollary 2 to find the vertices on the path, until we reach i. The formation of cycles is impossible, as this would imply a cyclic dependency structure, impossible for a regflow. We have to reach i as the set of vertices we choose from is finite. □

Note that there might be more than one stepwise influencing path from i to j. Next, I show that the structure of stepwise influencing paths imposes a strict restriction on the way a vertex on the stepwise influencing path can be connected.

Lemma 6 *Let $\wp_i(j)$ be a stepwise influencing path from i to j in an open graph (G, I, O) with flow $(f, \prec_f)$ and corresponding SSF $(s, \prec_s)$. Then $f^{-1}(j)$ is the only odd-placed vertex in $\wp_i(j)$ that j is connected to.*

Proof: According to the definition of stepwise influencing path, for every three consecutive vertices v_1, v_2, v_3 in $\wp_i(j)$ such that v_1 and v_3 are odd-placed we have that $v_2 = f(v_1)$ and $v_3 \in N(v_2) = N(f(v_1))$. According to Corollary 1 there must exist a Z-correction from v_1 to v_3. Therefore the odd-placed vertices in $\wp_i(j)$ are on a Z-path from i to $f^{-1}(j)$ and obviously from every odd-placed vertex in $\wp_i(j)$ there exists a Z-path to $f^{-1}(j)$. Lemma 3 says that j cannot be connected to any of the odd-placed vertices in $\wp_i(j)$. □

The previous lemma shows that the stepwise influencing paths can be used to describe some properties of the connectivity in open graphs with SSF. The next lemma (illustrated in Figure 23) will explain how a stepwise influencing path can be extended.

Lemma 7 *Let (G, I, O) be an open graph with regflow $(f, \prec_f)$ and corresponding SSF $(s, \prec_s)$ and let i and j be two non-output vertices of the open graph such that $f(j) \in s(i)$. If $v \in N(j) \cap s(i) \setminus \{f(j)\}$ then every stepwise influencing path $\wp_i(v)$ can be extended by the vertices j and $f(j)$ to create another stepwise influencing path $\wp_i(f(j))$.*

Proof: Adding j and $f(j)$ to $\wp_i(v)$ satisfies the conditions for stepwise influencing paths. There exists an edge between vertices j and v and vertices j and $f(j)$, hence it is a valid path. Moreover, $f(j) \in s(i)$ would be an even-placed vertex on the extended path, and j would be the unique oddly-placed vertex with $f(j) \in s(i)$. □

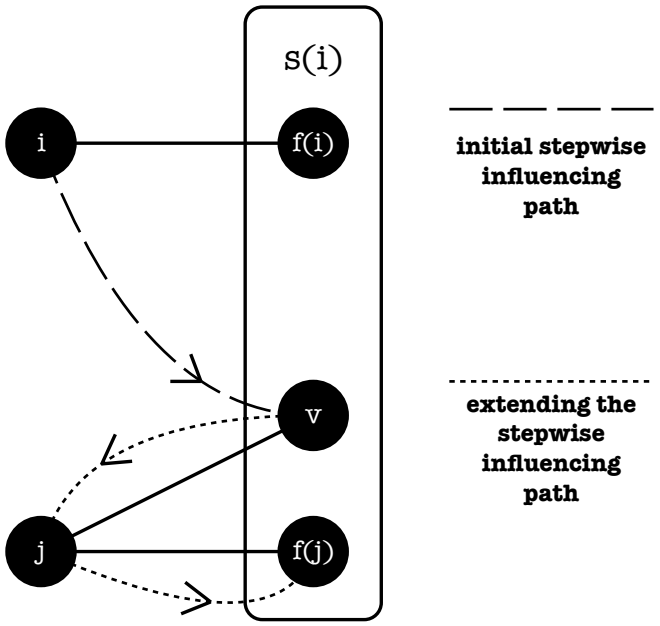


Figure 23: Extending a stepwise influencing path ending at vertex v according to Lemma 7.

The above lemmas will be used in Section 5.3.5 to obtain compact circuits from SSF.

5.3.4 Rewrite procedures for signal-shifted flow

In what follows we present the rewrite procedures (RPs) that will be used later in the compactification procedure (Algorithm 4). We refer to the i wire of each RP as the *target wire*, $j_1, ..., j_n$ as the *correcting wires* and finally k as the *neighbour wire*. Moreover, when we need to emphasise which RP we are referring to we also add a superscript to the wire label; for instance $k^{(2)}$ indicates the neighbour wire of RP2.

1. **Rewrite Procedure 2.** The rewrite procedure in Figure 24 moves gates $(E_{kj_1}, ..., E_{kj_m})$ past gates $(CX_{ij_1}, ..., CX_{ij_m})$, adding m many gates CZ_{ik} to slice $\mathcal{C}$ in the process. Using the rewrite rule in Figure 20-d, each of those E gates can be moved past gates $(CX_{ij_1}, ..., CX_{ij_m})$ in $\mathcal{C}$, creating a new CZ_{ik} each time the rule is applied (Figure 24-c).

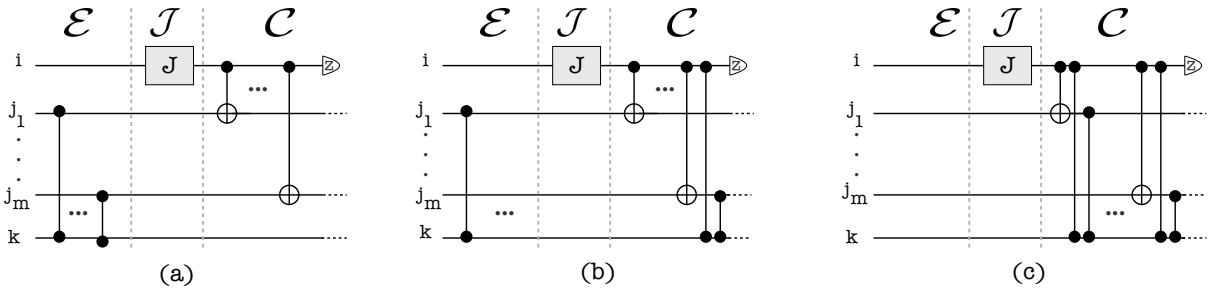


Figure 24: Rewrite Procedure 2. This RP is composed by several applications of the rewrite rule in Figure 20-d. Although all E gates are drawn in the $\mathcal{E}$ slice, the requirement for this RP to be applied is that those E gates are placed just before the corresponding CX gates (that is, with no gate in between).

2. **Rewrite Procedure 3.** The rewrite procedure in Figure 25 replaces gates $(E_{kj_1}, ..., E_{kj_{n-1}})$ in slice $\mathcal{E}'$ with $(CX_{j_1j_n}, CX_{j_2j_n}, ..., CX_{j_{n-1}j_n})$ in slice $\mathcal{C}$, removing gate CX_{ij_n} in the process. We use the rewrite rule in Figure 20-b for each pair $\{(E_{j_1k}E_{kj_n}), ..., (E_{j_{n-1}k}E_{kj_n})\}$ transforming gates $(E_{kj_1}, ..., E_{kj_{n-1}})$ into $(CX_{j_1j_n}, ..., CX_{j_{n-1}j_n})$, as depicted in Figure 25-b. The new CX gates can be pushed forward to the beginning of slice $\mathcal{C}$, since it commutes trivially with CX_{kj_n}. Using the rewrite rule in Figure 20-e we can commute $(CX_{j_1j_n}, ..., CX_{j_{n-1}j_n})$ past $(CX_{ij_1}, ..., CX_{ij_{n-1}})$ creating $(n-1)$ many new CX_{ij_n} in the process, which together with the pre-existing CX_{ij_n} in $\mathcal{C}$ will cancel out (since n is even), resulting in the circuit depicted in Figure 25-c.

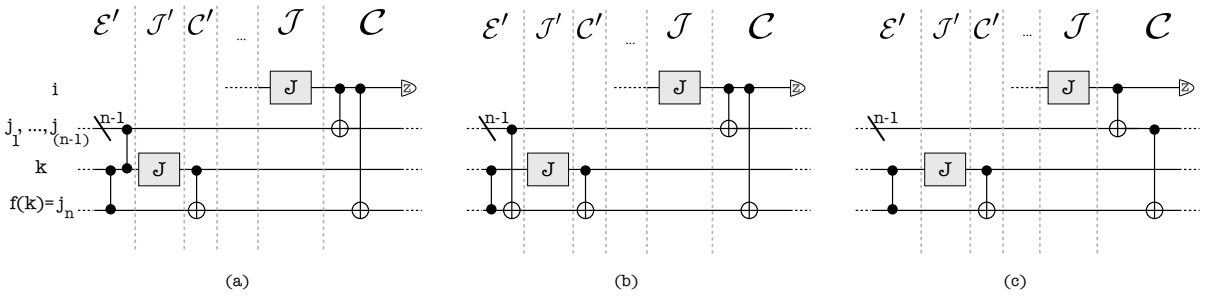


Figure 25: Rewrite Procedure 3. If $L_s(k) = L_s(i)$, slices $(\mathcal{E}, \mathcal{J}, \mathcal{C})$ and $(\mathcal{E}', \mathcal{J}', \mathcal{C}')$ become the same; The rewrite procedure remains exactly the same.

3. **Rewrite Procedure 4.** The rewrite procedure in Figure 26 replaces gates $(E_{kj_1}, ..., E_{kj_n})$ in slice $\mathcal{E}'$ with $(CX_{j_1f(k)}, ..., CX_{j_nf(k)})$ in slice $\mathcal{C}$. We use the rewrite rule in Figure 20-b for each pair $\{(E_{j_1k}E_{kf(k)}), ..., (E_{j_nk}E_{kf(k)})\}$ transforming gates $(E_{j_1k}, ..., E_{j_nk})$ into $(CX_{j_1f(k)}, ..., CX_{j_nf(k)})$ (Figure 26-b). The new CX gates can be pushed forward to the beginning of slice $\mathcal{C}$, since it commutes trivially with $CX_{kf(k)}$. Using the rewrite rule in Figure 20-e we can commute $(CX_{j_1f(k)}, ..., CX_{j_nf(k)})$ past $(CX_{ij_1}, ..., CX_{ij_n})$, creating n many $CX_{if(k)}$ in the process. Since n is even, all

those CX gates will cancel, resulting in the circuit depicted in Figure 26-c.

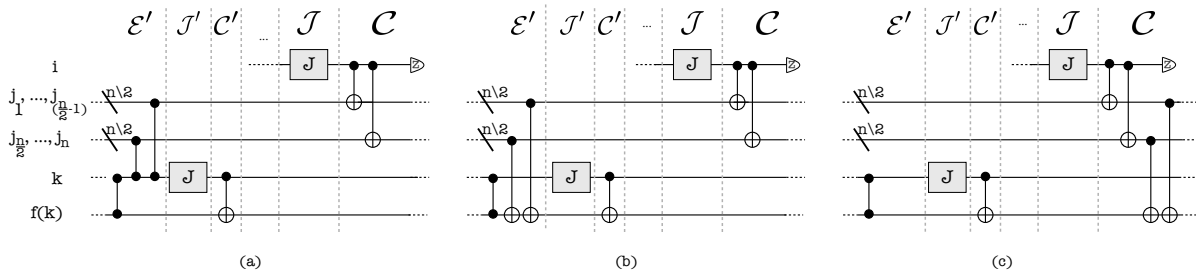


Figure 26: Rewrite Procedure 4. If $L_s(k) = L_s(i)$, slices $(\mathcal{E}, \mathcal{J}, \mathcal{C})$ and $(\mathcal{E}', \mathcal{J}', \mathcal{C}')$ become the same; The rewrite procedure remains exactly the same.

In order to apply the J-gate identity for the pair of wires $(i, f(i))$ of a SSF extended circuit, we need to rewrite it until all conditions in Definition 17 are satisfied. In a SSF extended circuit, the first two conditions are trivially satisfied for any pair of wires $(i, j) = (i, f(i))$ and, therefore, we need to rewrite the circuit to satisfy the other conditions. To do so we analyse each qubit k in the neighbourhood of $s(i)$, classifying it according to three different cases: (i) $L_s(k) < L_s(i)$, (ii) $L_s(k) \geq L_s(i)$ and $f(k) \in s(i)$ and (iii) $L_s(k) \geq L_s(i)$ and $f(k) \notin s(i)$. This separation into cases is necessary for two reasons: First, the distinction between $L_s(k) < L_s(i)$ and $L_s(k) \geq L_s(i)$ is necessary because we are interested in keeping the J-gate parallelization introduced by signal shifting and hence we need a different procedure to deal with each case. Secondly, in the case where $L_s(k) \geq L_s(i)$, we use the rewrite rule in Figure 20-b which deletes E gates. Since condition (ii) in Definition 17 requires the existence of gates of form $E_{kf(k)}$, we will treat differently cases where $f(k) \in s(i)$ and $f(k) \notin s(i)$ to guarantee those E gates will not be removed from the circuit. As we show next, for each case one of the RPs can be applied if a set of prior conditions are satisfied.

Proposition 4 *Let i be a wire in a SSF extended circuit s.t. J_i is in some slice $\mathcal{J}_n$. If there exists a wire k s.t. (i) $L_s(k) < L_s(i)$ and (ii) the set of gates $\{E_{kj_1}, ..., E_{kj_m}\}$ (with $j_1, ..., j_m \in s(i)$) can be pushed to slice $\mathcal{J}_n$, then RP2 (Figure 24) can be applied.*

Proof. Since $L_s(k) < L_s(i)$, the gate J_k belongs to a future slice $\mathcal{J}_m$ $(m > n)$. Also, $j_1, ..., j_m \in s(i)$ implies $i \prec_s \{j_1, ..., j_m\}$ and hence the gates $J_{j_1}, ..., J_{j_m}$ are in slices after $\mathcal{J}_n$ as well. Moreover, $j_1, ..., j_m \in s(i)$ implies there exist operators $X_{j_1}^{s_i}, ..., X_{j_m}^{s_i}$ in the measurement pattern, which are translated to the extended circuit as a $CX_{ij_1}, ..., CX_{ij_m}$ in slice $c_{n,i}$ (according to Definition 14). Thus, if every gate E_{kj}, $j \in s(i)$, can be trivially

pushed to slice $\mathcal{J}_n$, we have exactly the scenario depicted in Figure 24-a. Therefore, the rewrite procedure in Figure 24 can be applied. □

For reasons that will become clear in Sec. 5.3.5, where the algorithm to obtain compact circuits from SSF extended circuits is introduced, we need to consider a case which is slightly different from the scenario described in Proposition 4. In this case, condition (ii) in Proposition 4 is not satisfied because some of the E gates are in slice $c_{n,i}$ but cannot be pushed trivially back to $\mathcal{J}_n$. The only scenario where that could happen is if there exists $CX_{i,k}$ in slice $c_{n,i}$ and some of the E_{kj} gates are placed past it (and hence cannot be pushed trivially to $\mathcal{J}_n$). In this scenario, as we show next, RP2 can also be applied.

Proposition 5 *Let i be a wire in a SSF extended circuit s.t. J_i is in some slice $\mathcal{J}_n$. If there exists a wire k s.t. (i) $k \in s(i)$ and (ii) gates $\{E_{kj_1}, ..., E_{kj_m}\}$ (with $j_1, ..., j_m \in s(i)$) can either be all pushed to slice $\mathcal{J}_n$ or just some can be pushed to slice $\mathcal{J}_n$ and the other E gates are placed in $c_{n,i}$ just after CX_{ik}, then RP2 (Figure 24) can be applied.*

Proof. The proof of this proposition is simple due to its similarity to Proposition 4. First, note that $k \in s(i)$ implies $L_s(k) < L_s(i)$ and hence condition (i) in this proposition is equivalent to the one in Proposition 4. Therefore, if all gates in the set $\{E_{kj_1}, ..., E_{kj_m}\}$ can be pushed to slice $\mathcal{J}_n$ we have exactly the conditions in Proposition 4 and there is nothing to prove. The other possibility is when a subset of the set $\{E_{kj_1}, ..., E_{kj_m}\}$ can be pushed to slice $\mathcal{J}_n$ but gates in the complementary subset are placed in slice $c_{n,i}$, after gate CX_{ik} (which exists since $k \in s(i)$). It is easy to note that it will not prevent the application of RP2, since the only gate in $c_{n,i}$ that could be placed in between the E and CX gates used in RP2, namely CX_{ik}, is assumed in condition (ii) to be placed before the gates of the form E_{kj}. □

In the next Lemma we show the interesting effect of applying RP2 to SSF extended circuits.

Lemma 8 *The application of RP2 to a pair of target and neighbour wires (i, k) of a SSF extended circuit removes all CZ_{ik} from the circuit.*

Proof. According to Definition 14, all CZ gates with control in wire i are placed in slice $c_{n,i}$ and hence we only need to show that all CZ_{ik} in that slice are removed. Let us divide the analysis into two cases: (i) $k \in Odd(s(i))$ and (ii) $k \notin Odd(s(i))$. Consider the first case. Since $k \in Odd(s(i))$, there exists a $Z_k^{s_i}$ in the measurement pattern and

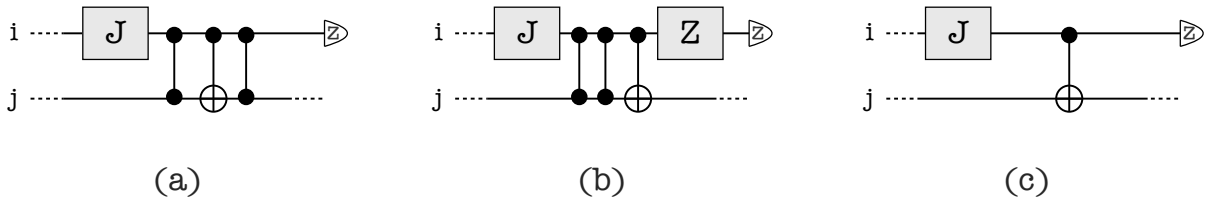


Figure 27: Removing even many CZ_{ij} from a SSF extended circuit. See Lemma 8 for more information.

hence there exists a gate CZ_{ik} in slice $c_{n,i}$. For this case, the index m in RP2 is an odd number and therefore the application of RP2 creates odd many CZ_{ik} gates in slice $c_{n,i}$. Note that those CZ_{ik} gates can be created in different parts of $c_{n,i}$ (depending whether Proposition 4 or 5 is satisfied) such that they cannot be grouped together trivially (like in Figure 27-a). It is easy to verify that the only possible scenario for that to happen is when there exists CX_{ik} in $c_{n,i}$ in between the created CZ_{ik} gates. However, those CZ_{ik} gates can be grouped together by moving all CZ_{ik} to one side of the troublesome CX_{ik} using the identity

$$CZ_{ik}CX_{ik}|\psi\rangle_i|\phi\rangle_k = CX_{ik}CZ_{ik}(Z_i|\psi\rangle_i)|\phi\rangle_k \tag{5.21}$$

where $|\psi\rangle$ and $|\phi\rangle$ are arbitrary quantum sates and Z is the Pauli operator (see Figure 27-b). This way, since there exist even many CZ_{ik} in sequence and even many CZ gates equal the identity, all CZ_{ik} can be simply removed from the circuit. The created single-qubit gate Z_i can be pushed forward to the end of the i wire, since it commutes with all gates in $c_{n,i}$. Moreover, since the final measurement is onto the Z-basis, the operator Z_i has no effect in the measurement statistics and therefore can be removed from the circuit without changing the computation being implemented by the circuit (Figure 27-c). An equivalent analysis applies for the second case, where $k \notin Odd(s(i))$. For this case RP2 create even many CZ_{ik} in slice $c_{n,i}$ and there is no pre-existing CZ_{ik} in that slice. The same identity can be used to group together all created CZ_{ik}, which cancel out since there are even many of them. Since both created and pre-existing CZ_{ik} are removed from the circuit by the application of RP2, the Lemma holds. □

Now we analyse the cases where $L_s(k) \geq L_s(i)$, that is, with J_k gate blocking all gates of form E_{kj} to be pushed past the correction gates (necessary to satisfy condition (*iii*) in Definition 17). Since in those cases the corresponding RP will delete E gates and we do not want to delete E gates of the form $E_{i,f(i)}$ (see Definition 17), we analyse separately the case where $f(k) \in s(i)$ (Proposition 6) and the case where $f(k) \notin s(i)$ (Proposition 7).

Proposition 6 *Let i be a wire in a SSF extended circuit s.t. J_i is in slice $\mathcal{J}_n$. If there exists a wire k such that the following conditions are satisfied: (i) $L_s(k) \geq L_s(i)$; (ii) $f(k) \in s(i)$; (iii) gates $E_{kj_1}, ..., E_{kj_n}$ (with $j_1, ..., j_n \in s(i)$) can be trivially pushed to some slice $\mathcal{J}_{n'}$ (containing J_k); and (iv) the $E_{kf(k)}$ gate is the first gate acting on wire $f(k)$, then RP3 (Figure 25) can be applied.*

Proof. Since $L_s(k) \geq L_s(i)$, $n' \leq n$. In slice $\mathcal{C}_{n'}$ there is a $CX_{kf(k)}$; since $f(k) \in s(i)$, we define $j_n = f(k)$. Also, $j_1, ..., j_n \in s(i)$ implies that $i \prec_s \{j_1, ..., j_n\}$ and hence the gates $J_{j_1}, ..., J_{j_n}$ are in some future slices (compared to n). Moreover, it also implies the existence of the operators $X^{s_i}_{j_1}, ..., X^{s_i}_{j_n}$ in the measurement pattern, which are translated to the extended circuit as $CX_{ij_1}, ..., CX_{ij_n}$ in slice $c_{n,i}$ (containing all the CX and CZ with control on qubit i). On the other hand, $L_s(k) \geq L_s(i)$ implies both $k \notin Odd(s(i))$ and $k \notin s(i)$ and therefore there is no CZ_{ik} or CX_{ik} in the circuit. Thus, if every gate $E_{kj_1}, ..., E_{kj_n}$ can be trivially pushed to slice $\mathcal{J}_{n'}$ and $E_{kf(k)}$ is the first gate acting on wire j_n, we have exactly the scenario depicted in Figure 25-a. Therefore, the circuit identity in Figure 25 can be applied. □

Proposition 7 *Let i be a wire in a SSF extended circuit s.t. J_i is in slice $\mathcal{J}_n$. If there exists a wire k such that the following conditions are satisfied: (i) $L_s(k) \geq L_s(i)$; (ii) $f(k) \notin s(i)$; (iii) gates $E_{kj_1}, ..., E_{kj_n}$ (with $j_1, ..., j_n \in s(i)$) can be trivially pushed to slice $\mathcal{J}_{n'}$ (containing J_k); and (iv) the $E_{kf(k)}$ gate is the first gate acting on wire $f(k)$, then RP4 (Figure 26) can be applied.*

Proof. Since $L_s(k) \geq L_s(i)$, then $n' \leq n$. In slice $\mathcal{C}_{n'}$ there is a $CX_{kf(k)}$, with $f(k) \notin s(i)$. Also, $j_1, ..., j_n \in s(i)$ implies that $i \prec_s \{j_1, ..., j_n\}$ and hence the gates $J_{j_1}, ..., J_{j_n}$ are also in some future slices (compared to n). Moreover, it also implies the existence of the operators $X^{s_i}_{j_1}, ..., X^{s_i}_{j_n}$ in the measurement pattern, which are translated to the extended circuit as $CX_{ij_1}, ..., CX_{ij_n}$ in slice $c_{n,i}$ (containing all the CX and CZ with control on qubit i). On the other hand, $L_s(k) \geq L_s(i)$ implies both $k \notin Odd(s(i))$ and $k \notin s(i)$ and therefore there is no CZ_{ik} or CX_{ik} in the circuit. Thus, if every gate $E_{kj_1}, ..., E_{kj_n}$ and $E_{kf(k)}$ can be trivially pushed to slice $\mathcal{J}_{n'}$ and $E_{kf(k)}$ is the first gate acting on wire $f(k)$, we have exactly the scenario depicted in Figure 26-a. Therefore, the circuit identity in Figure 26 can be applied. □

In what follows we analyse some properties regarding the interplay between the RPs that will be crucial for the compactification algorithm for SSF extended circuits.

Lemma 9 *Let C be an extended circuit obtained from a signal shifted measurement pattern. Then, if a rewrite procedure removes a gate E_{jk} from the circuit, it will not be required for the application of any other rewrite procedure.*

Proof. Note that only RP3 and RP4 can delete a E_{jk} gate from the circuit. The proof is divided into three parts: (i) we show that the E_{jk} is never a flow edge and, therefore, it will not be required for the application of a J-gate identity; (ii) it will not be needed for the application of RP2 and (iii) it will not be needed to create a CX (by using RP3 or RP4) in the next layers. Suppose E_{jk} is a flow edge such that $k = f(j)$, with j being any of qubits $(j_1, ..., j_{n-1})$ in RP3 or $(j_1, ..., j_n)$ in RP4. The other way around, that is, $j = f(k)$, is not possible since $f(k)$ is separately identified in both RP3 and RP4. Since $f(j) \in s(j)$ for all $j \in O^C$, we have $j \prec_s k$. By construction, RP3 and RP4 are applied if and only if $k \preccurlyeq_s i$. Moreover, since $j \in s(i)$ in both RP3 and RP4, it holds that $i \prec_s j$. Putting everything together we have $i \prec_s j \prec_s k \preccurlyeq_s i$, which is a contradiction. Therefore, E_{jk} is never a flow edge.

In this second part of the proof, we show that if E_{jk} is deleted by RP3 or RP4, it will not be required for the application of RP2 in any future step. Using the notation defined earlier, the fact that RP2 is applied after RP3 or RP4 can be written as $i^{(3)} \prec_s i^{(2)}$ or $i^{(4)} \prec_s i^{(2)}$, where i is the target wire of the corresponding RP. This relation will be used in the rest of the proof. We show that $k^{(3)}$ and $k^{(4)}$ cannot be any of wires $j_1^{(2)}, ..., j_n^{(2)}$ (denoted simply by $j^{(2)}$ in the rest of this proof) or $k^{(2)}$ and hence the gates $E_{j^{(3)}k^{(3)}}$ or $E_{j^{(4)}k^{(4)}}$ cannot be the same as $E_{j^{(2)}k^{(2)}}$, which is what we want to prove. Let us show for $k^{(3)}$. Since $k^{(3)} \preccurlyeq_s i^{(3)}$ and $i^{(2)} \prec_s k^{(2)}$ (true for both Propositions 4 and 5, which state the scenarios where RP2 is applied) we have $k^{(3)} \preccurlyeq_s i^{(3)} \prec_s i^{(2)} \prec_s k^{(2)}$ and therefore $k^{(3)}$ cannot be the same wire as $k^{(2)}$. Now assume wire $j^{(3)}$ is $k^{(3)}$. In RP4 we have $k^{(4)} \preccurlyeq_s i^{(4)}$ (by construction) and in RP2 it holds that $i^{(2)} \prec_s j^{(2)}$, since the target of a correcting gate is always placed before the J-gate acting on that same wire (Definition 14). Putting everything together gives $k^{(3)} \preccurlyeq_s i^{(3)} \prec_s i^{(2)} \prec_s j^{(2)}$. But since by assumption $j^{(2)}$ is $k^{(3)}$, it gives $k^{(3)} \prec_s k^{(3)}$ which is a contradiction. The proof is the same for $k^{(4)}$. Therefore, if E_{jk} is "consumed" by RP3 or RP4, it will not be required for the application of RP2 in any future step.

Finally for the third part note that an E gate will be removed to create a CX gate only through identity in Figure 20-b. The pair of such E gates is always associated with a flow and a non-flow edge when used in RP3 or RP4. Therefore, there are two possible pairs of E gates using the non-flow edge E_{jk}: $E_{jk}E_{kf(k)}$ (which would create $CX_{jf(k)}$) and

$E_{jk}E_{jf(j)}$ (which would create $CX_{kf(j)}$). We want to show that whenever we "consume" E_{jk}, it will not be required to create a different CX than the one already created. Suppose we use E_{xy} to create $CX_{xf(y)}$. This is the case when $y \preccurlyeq_s i$ and $i \prec_s x$, if x is a correcting wire and y is a neighbour wire in either RP3 or RP4. Similarly, if x is the neighbour wire and y is the correcting one then $x \preccurlyeq_s i$ and $y \succ_s i$ (easily obtained by relabelling the wires in RP3 and RP4). Since the first pair of conditions is inconsistent with the last pair of conditions ($j \preccurlyeq_s i$ and $k \succ_s i$), we conclude that once a given E is used to create a CX, that E would not be required to create a different CX in any further step of the algorithm since the partial order $\prec_s$ is not changed by the application of any RP. □

Corollary 3 *Let CX_{ij} be a gate in an extended circuit created by the application of a rewrite procedure. The E gate "consumed" to create this CX_{ij} can be univocally determined.*

Proof. It follows from the third part of the proof of Lemma 9 that if a gate of the form E_{jk} is deleted in RP3 or RP4 we will have one of the following cases: the gates $E_{jk}E_{kf(k)}$ are used to create $CX_{jf(k)}$ or the pair $E_{jk}E_{jf(j)}$ are used to create $CX_{kf(j)}$. The CX that is created by "consuming" the E_{jk} depends exclusively on the relation between vertices j, k and $f(j)$ in respect to the partial order $\prec_s$. Since the partial order $\prec_s$ is not changed by the application of any RP, there is a one-to-one correspondence between the deleted E gate and the newly created CX gate. □

The process of choosing the correct RP to be applied is summarized in Algorithm 3, which will be used as a subroutine in the compactification algorithm for SSF extended circuits (Algorithm 4). The modified conditions in Lines 5 and 7 are based on the fact that if a CZ required for the application of RP3 or RP4 is not in the circuit, the CX that it would create will be (see Lemma 9), allowing the application of future RPs. In all RPs we assumed that some E gates could be moved to the beginning of a given slice $\mathcal{J}$ in a trivial way. It will not be true in general for SSF extended circuits, since there might exist several other gates in the extended circuit such that the aforementioned E gates could not be moved trivially to $\mathcal{J}$. In other words, the conditions in Propositions 4 to 7 would not be satisfied. The algorithm in the next section addresses exactly this problem: it provides an ordering where Algorithm 3 never aborts. This ordering is the *global* structure coming into play, since it is related to SSF and flow of the graph.

Algorithm 3: This algorithm decides which Rewrite Procedure must be applied by analysing how a given qubit k is connected with the correcting set of another qubit i

Input: Wire labels i and k.
Output: A description of which Rewrite Procedure must be applied.

begin
  Read i and k.
  if *all conditions in Proposition 4* ***or*** *Proposition 5 are satisfied* **then**
    Apply RP2;
  if *all conditions in Proposition 6 are satisfied* ***or*** *conditions (iii) and/or (iv) in Proposition 6 are not satisfied but there exist a sequence of* $(CX_{j_1 j_n}, ..., CX_{j_{n-1} j_n})$ *gates that can be pushed trivially to* $c_{n,i}$ **then**
    Apply RP3;
  if *all conditions in Proposition 7 are satisfied* ***or*** *conditions (iii) and/or (iv) in Proposition 7 are not satisfied but there exist a sequence* $(CX_{j_1 f(k)}, ..., CX_{j_n f(k)})$ *gates that can be pushed trivially to* $c_{n,i}$ **then**
    Apply RP4;
  otherwise Abort

5.3.5 Compactification procedure for signal-shifted flow

In this section I explain how the compactification algorithm for SSF extended circuits (Algorithm 4) works. The goal of Algorithm 4 is to create $|O^C|$ many J-blocks (see Definition 17) in a SSF extended circuit and then apply the J-gate identity for all J-blocks, removing $|O^C|$ many wires from the extended circuit. Therefore, the output of Algorithm 4 is the compact form of the inputed SSF extended circuit. One of the main differences between SSF and other gflows is the notion of stepwise influencing paths, introduced in Section 5.3.3. The next lemma uses some properties of stepwise influencing paths to relate the SSF correcting set to the partial order of regflow. This relation will play an important role in Algorithm 4 since it gives an appropriate ordering for the application of the RPs.

Lemma 10 *Let* (G, I, O) *be an open graph with regular flow* $(f, \prec_f)$ *and SSF* $(s, \prec_s)$. *Then, for all* $v \in N(j) \setminus \{i\}$, *where* $j \in s(i)$, *it holds that* $v \succ_f i$.

Proof. First suppose $j = f(i)$. Then, by regflow definition (Def. 10), $v \succ_f i$. Now suppose $j \neq f(i)$. Then from Def. 10, $v \succ_f f^{-1}(j)$. By Lemma 5, there exists a step-wise influencing path passing through $f(i)$ and $f^{-1}(j)$, namely $\wp_i(j)$. It follows from Corollary 2, that there exists $f^{-1}(k)$ such that $k \in s(i)$ and $f^{-1}(j) \in N(k)$ and, consequently, $f^{-1}(j) \succ_f f^{-1}(k)$. Lemma 7 allows us to repeat this process to find the previous vertices

in the path $\wp_i(j)$ until we reach vertex i. Hence we conclude that $f^{-1}(j) \succ_i i$ and therefore $v \succ_f f^{-1}(j) \succ_f i$. $\square$

Before running Algorithm 4, the slices $c_{n,i}$ must be arranged in the extended circuit from right to left respecting the order imposed by $\prec_f$. This can always be done since the set of gates in a given slice $c_{n,i}$ commutes with the gates in an other slice $c_{n,j}$, for any valid j. The algorithm starts with a `for` loop that runs for one SSF layer at a time, starting with the input layer and moving onwards until the last layer of non-output qubit is considered. At each iteration n of the `for` loop, a rewrite procedure is applied to each pair of qubits (i,k) s.t. J_i is in $\mathcal{J}_n$ and $k \in N(s(i))$. The two `foreach` loops and the two `while` loops define the order in which Algorithm 3 will be called. This ordering, which is the inverse of the order given by $\prec_f$, assures that Algorithm 3 will never abort when it is called by Algorithm 4.

Algorithm 4: Transforms a SSF extended circuit into a compact circuit.

Input: SSF extended circuit.
Output: Compact form of the SSF extended circuit.

begin
  $D \leftarrow \max_{i \in G}\{L_s(i)\}$
  for $n = 1$ *to* D **do**
    Let $W_n = \{a_{n,1}, ..., a_{n,m_n}\}$ be the set of the m_n wires with J-gate in slice $\mathcal{J}_n$
    Let $S^n_{max} \leftarrow \max_{i \in W_n}\{L_f(i)\}$
    Let $S^n_{min} \leftarrow \min_{i \in W_n}\{L_f(i)\}$
    while $S^n_{max} \geq S^n_{min}$ **do**
      foreach $a_{n,i} \in W_n$ *such that* $L_f(a_{n,i}) = S^n_{max}$ **do**
        Let $NSTEP \leftarrow \max_{k \in \{N(s(a_{n,i})) \setminus \{a_{n,i}\}\}}\{L_f(k)\}$
        while $NSTEP > S^n_{max}$ **do**
          `// That is, for all` $k \in \{N(s(a_{n,i})) \setminus \{a_{n,i}\}\}$ `(Due to Lemma 10)`
          foreach $k \in \{N(s(a_{n,i})) \setminus \{a_{n,i}\}\}$ *such that* $L_f(k) = NSTEP$ **do**
            $i \leftarrow a_{n,i}$;
            Run Algorithm 3
          $NSTEP \leftarrow (NSTEP - 1)$
      $S_{max} \leftarrow (S_{max} - 1)$
  Remove from the circuit all gates CX_{ij} placed in any slice $\mathcal{C}_n$ such that $j \neq f(i)$.
  Apply the J-gate identity for all pairs of wires $\{i, f(i)\}$ such that $i \in O^C$.

Lemma 11 *Algorithm 3 never aborts when called by Algorithm 4.*

Proof. Let us start by showing that the algorithm works for the first iteration of the `for` loop (*i.e.*, $n = 1$) and then we explain why it will work for all other layers. In what follows we use the notation $a_{n,i}$ to represent a target wire i in layer n; note that there might exist more than one such wire in the same layer. Let S^n_{max} (S^n_{min}) be the maximum (minimum) value of $L_f(a_{n,i})$ for $a_{n,i} \in W_n$ (as defined in Lines 5 and 6 in the algorithm), where W_n is the set of the m_n wires with J-gate in slice $\mathcal{J}_n$. According to Lemma 10, for any $a_{1,i}$ such that $L_f(a_{1,i}) = S^1_{max}$, and any qubit k connected to a qubit in $s(a_{1,i})$ it holds that $k \notin W_1$. In this case, condition (i) in Proposition 4 is satisfied. Since we are in the first SSF layer, all E gates required for the application of any of those RPs can be trivially pushed forward to slice $\mathcal{J}_1$, hence all conditions in Proposition 4 are initially satisfied. After the application of one or more RP2, it might happen that some E gates get stuck in between two CX gates in slice $c_{1,i}$ (which happens when a neighbour wire k is itself in both $N(s(a_{1,i}))$ and $s(a_{1,i})$ sets). In this case the conditions in Proposition 5 are then satisfied. In both cases, only RP2 can be selected by Algorithm 3. Therefore, Algorithm 3 would not abort for all target qubits $a_{1,i}$ such that $L_f(a_{1,i}) = S^1_{max}$. As a consequence, since only RP2 is applied, all gate E_{kp}, such that $p \in s(a_{1,i})$, are moved past the correction slice $c_{1,i}$, for all $a_{1,i}$ such that $L_f(a_{1,i}) = S^1_{max}$.

Now consider qubits $a_{1,i}$ such that $L_f(a_{1,i}) = S^1_{max} - 1$, that is, the second iteration of the first `while` loop. For $k \in N(s(a_{1,i}))$, Lemma 10 implies that either $k \notin W_1$ or $\{(k \in W_1) \wedge [L_f(k) = S^1_{max}]\}$ is true. For $k \notin W_1$, the procedure to be applied is RP2, similarly and by the same reasons as in the previous iteration. For the other case the applicable rewrite procedures are either RP3 or RP4, since J_k in $\mathcal{J}_1$ implies $L_f(k) = L_f(a_{1,i})$. Note that in the previous iteration (qubits $a_{1,i}$ s.t. $L_f(a_{1,i}) = S^1_{max}$, like the wire k being analysed now), all $E_{f(k)v}$, for any $v \neq k$ were moved past the corresponding correction slice and hence $E_{f(k)k}$ is the first gate acting on the $f(k)$ wire (as required by both Propositions 6 and 7). Moreover, when the circuit identity in Figure 20-b creates the new CX gates (first step in both RP3 and RP4), it can be pushed forward to $c_{1,i}$ because the only gate in between is $CX_{kf(k)}$, which commutes with the CX gates we want to push forward. Therefore, since all conditions in either Proposition 6 or Proposition 7 would be satisfied for all qubits $a_{1,i}$ such that $L_f(a_{1,i}) = S^1_{max} - 1$, the corresponding RP can be successfully applied.

To complete the proof by induction for the first iteration of the `for` loop, assume Algorithm 3 has not aborted in the first q iteration of the first `while` loop. Then in the $(q+1)^{th}$ iteration, qubits $a_{1,i}$ such that $L_f(a_{1,i}) = (S^1_{max} - q) \geq S^1_{min}$, where $S^1_{min} \equiv \min_{a_{1,i} \in W_1}\{L_f(a_{1,i})\}$, are considered. If a given qubit k is neighbour of a qubit in $s(a_{1,i})$,

then Lemma 10 implies that either $k \notin W_1$ or $\{(k \in W_1) \wedge [(S^1_{max} - q) < L_f(k)\}$ is true. Here the same analysis as before applies; First, if $k \notin W_1$, Algorithm 3 will apply RP2, moving the corresponding E gates past slice $c_{1,i}$. On the other hand, if $\{(k \in W_1) \wedge [(S^1_{max} - q) < L_f(k)\}$, either the conditions in Proposition 6 or Proposition 7 will be satisfied, since all E gates acting on qubits in the sets $s(a_{1,v})$, for all $a_{1,v} \in W_1$ s.t. $L_f(a_{1,v}) > (S^1_{max} - q)$, were moved past slice $c_{1,v}$ in previous iterations (with the obvious exception of gates of the form $E_{vf(v)}$). The procedure continues until there is no qubit left in W_1 to be considered. Therefore, Algorithm 3 never aborts when called during the first iteration ($n = 1$).

Now let us analyse how the E gates placement changed in the circuit after the n^{th} iteration of the `for` loop where Algorithm 3 applied a RP each time it was called by Algorithm 4. First, note that all E gates acting on qubits in $s(i)$, for $i \in W_n$, were moved to $\mathcal{E}_{n+1}$ (via RP2) or transformed into a CX (via RP3 or RP4) and then moved to the same slice, with the exception of gates of the form $E_{if(i)}$ which remains in $\mathcal{E}_n$ and will later be used for the application of the J-gate identity. Moreover, Lemma 9 guarantees that if an E gate is deleted in a given iteration of the algorithm, it will not be required in any future step of the algorithm.

Note also that those E and CX gates were moved to the $\mathcal{E}_{n+1}$ slice in a specific order, namely the inverse of the order induced by $\prec_f$. Since this ordering is a property of the associated graph, it does not change during the run of the algorithm. It means that for any given iteration n of the `for` loop, the ordering of the two-qubit gates required for the application of a RP will be in agreement with the order Algorithm 3 will be called by Algorithm 4. Hence the condition (required for the application of any RP) that the required two-qubit gates can be pushed trivially to a given slice is always satisfied. Therefore, at each iteration of the `for` loop, it arranges all two-qubit gates necessary for the next iteration in the same order it will be called.

To complete the proof by induction, let us assume that in the first $(p-1)$ iterations of the `for` loop Algorithm 3 has not aborted, and then we show that it will not abort in the p^{th} iteration. Let us analyse the first iteration of the first `while` loop for $n = p$. According to Lemma 10, for any $a_{p,i}$ such that $L_f(a_{p,i}) = S^p_{max}$, if a given qubit $k \in N(s(a_{p,i}))$ then either $k \in W_m$ (if there exists $m < p$ s.t. $S^m_{max} > S^p_{max}$) or the J_k gate belongs to a layer $\mathcal{J}_{m'}$ s.t. $m' > p$. If the latter condition is the case, then by definition only RP2 is applied to $a_{p,i}$. On the other hand, if $k \in W_m$, RP3 or RP4 must be applied. Since by assumption Algorithm 3 has not aborted in any previous iteration, the conditions for the application

of the aforementioned RPs are satisfied and Algorithm 2 will not abort in the current iteration.

Now assume Algorithm 3 has not aborted in the first q iteration of the first `while` loop for $n = p$. In the $(q+1)^{th}$ iteration of this loop, qubits $a_{p,i}$ such that $L_f(a_{p,i}) = (S^p_{max} - q) \geq S^p_{min}$, where $S^p_{min} \equiv \min_{a_{p,i} \in W_p}\{L_f(a_{p,i})\}$, are considered. For a given k neighbour of a qubit in $s(a_{p,i})$, Lemma 10 implies that one of the following statement holds: (i) $k \in W_m \wedge [S^m_{max} > (S^p_{max} - q)]$ for some $m < p$, (ii) $(k \in W_p) \wedge [(S^p_{max} - q) < L_f(k)]$ or (iii) $k \in W_{m'}$, for $m' > p$. Here the same analysis as before applies; For cases (i) and (ii) RP3 or RP4 is applied and for (iii) RP2 will move the corresponding E gates past slice $c_{p,i}$. Since the same order is respected throughout the algorithm (the inverse of the order defined by flow), the conditions for the application of the aforementioned RPs are satisfied and Algorithm 2 will not abort in the current iteration. This procedure is repeated until there is no more elements in the set W_p, which concludes our proof by induction and proves the Lemma. □

Theorem 4 *Algorithm 4 outputs a compact circuit.*

Proof. We show that Algorithm 4 creates $|O^C|$ many J-gate blocks (Definition 17) and then removes $|O^C|$ many wires from the circuit (using the J-gate identity), yielding a compact circuit. The first two conditions in Definition 17 are trivially satisfied for all SSF extended circuits. In Lemma 11 we proved that Algorithm 4 moves to a future slice (or removes) all gates E acting on qubits $j \in s(i)$ $(\forall i \in O^C)$ except gate $E_{if(i)}$ and hence condition 3 in Definition 17 is also satisfied. Now we prove that condition 4 is satisfied. Note that all non-input qubits are initialized in state $|+\rangle$. According to the SSF construction (Proposition 3), for all qubit $j \in s(i)$, there exists $f^{-1}(j)$. By definition $f : O^C \to I^C$ and therefore every qubits in $s(i)$, for all $i \in O^C$, starts in the $|+\rangle$ state. Therefore, all remaining CX_{ij} such that $j \neq f(i)$ can me removed from the circuit without changing the computation, since $CX_{ij}|+\rangle_j = |+\rangle_j$ (remember that the only gate E acting on wires $j \in s(i)$ left behind by the algorithm is $E_{if(i)}$). Therefore, after the step in Line 17, condition 4 in Definition 17 is also satisfied. Since all non-output wires in an extended circuit are measured in the Z basis by construction, condition 5 is also satisfied and hence Algorithm 4 has created $|O^C|$ many J-blocks after the step in Line 17. Finally, in Line 18 the J-gate identity is applied to all J-blocks, resulting in a compact circuit. □

A step-by-step application of Algorithm 4 to an example is given in Chapter 6.

5.3.6 Discussion of results and possible extensions

In the last few sections I have introduced and explored the SSF. I have also developed an algorithm able to obtain a compact circuit from a SSF measurement pattern. In Chapter 6, I will use the SSF compactification procedure to propose an automated optimization procedure for quantum circuits: starting with a circuit, I will translate it to the 1WQC model as a regflow pattern, optimize it using signal shifting and then translate the SSF measurement pattern back to the circuit model using Algorithm 4. This automated optimization scheme explores the global structure of circuit to parallelize several J-gates and to group together several CX gates, which allows further optimizations to take place using specific method for Clifford gate parallelization.

Interestingly the scheme fails to compactify the parallel pattern obtained via Pauli simplification rules [23]. In other words one needs to keep the extra space to keep the parallel depth obtained due to the Pauli measurements. This further indicates the crucial role that Clifford computation (corresponding to Pauli measurements) plays in obtaining the superior parallel power of MBQC over quantum circuits. In the example below, I explore this time-memory tradeoff when Pauli measurements are considered. I give several circuits (implementing the same computation) from the same measurement pattern, varying the number of $\mathcal{J}$ layers and wires from one compact circuit to another.

Example: Pauli measurements

In this example we explore the role of Pauli measurements in the context of compactification procedures. Let us start by reviewing how the correction operators are modified when applied to a qubit measured with 0 and $\pi/2$ angles (corresponding to Pauli measurements):

$$M_i^{\frac{\pi}{2}} X_i^s = M_i^{\frac{\pi}{2}} Z_i^s \tag{5.22}$$

$$M_i^0 X_i^s = M_i^0 \tag{5.23}$$

Note that in the MBQC context, both substitutions clearly might reduce the depth of the computation: Equation 5.22 substitutes Z_i^s for X_i^s, which can then be removed using signal-shifting, and Equation 5.23 simply deletes the existing X_i^s dependency. However, in the associated extended circuit, these optimisations can not be implemented alongside the SSF compactification procedure developed in this Chapter (Algorithm 4), forcing one to choose between optimization in time or memory. To see why, note that Equations 5.22

and 5.23 both change the correcting structure for every qubit encoded in the index s. As a consequence, the step-wise influencing path $\wp$ (Definition 21), which is the backbone of the compactification protocol, can not be defined anymore for these qubits.

In summary, if one wants to save memory in the circuit model, the substitutions in Equations 5.22 and 5.23 must be avoided, allowing the identification of all step-wise influencing paths and, consequently, the removal of auxiliary qubits. Conversely, if J-gate parallelization is the goal, Algorithm 4 can be easily adapted to create just $|O^C| - p$ many J-blocks in the extended circuit (instead of $|O^C|$ many), where p is the number of Pauli measurements in the associated measurement pattern.

Consider the circuit shown in Figure 28-a. It has two J gates with arbitrary angles (arbitrary angles are omitted in the Figure) and two Pauli angles, namely 0 and $\frac{\pi}{2}$. Using the method described in Sec. 4.2.2, we can translate this circuit to the MBQC model; the associated graph is shown in Figure 28-b and the flow measurement pattern is given by:

$$Z_6^{s_4} Z_3^{s_1} X_6^{s_5} X_3^{s_2} M_5^0 M_2^{\pi/2} Z_2^{s_4} X_5^{s_4} M_4^{\theta_4} Z_5^{s_1} Z_4^{s_1} X_2^{s_1} M_1^{\theta_1} E_G \tag{5.24}$$

where, $E_G = E_{56}E_{45}E_{25}E_{24}E_{23}E_{12}$. Using the signal shifting technique (Sec. 5.3.1), we obtain the following SSF measurement pattern:

$$Z_6^{s_1+s_4} Z_3^{s_1} X_6^{s_5+s_1} X_3^{s_1+s_2+s_4} M_5^0 M_2^{\pi/2} X_2^{s_1} X_5^{s_1+s_4} M_4^{\theta_4} M_1^{\theta_1} E_G \tag{5.25}$$

Now let us see what happens when we use the fact that some measurements are Pauli measurements. Using Equation 5.22, we can optimize the pattern by removing the dependency between measurements 1 and 2, obtaining the following measurement pattern:

$$Z_6^{s_1+s_4} Z_3^{s_1} X_6^{s_5+s_1} X_3^{s_2+s_4} M_5^0 X_5^{s_1+s_4} M_2^{\pi/2} M_4^{\theta_4} M_1^{\theta_1} E_G \tag{5.26}$$

Alternatively, the measurement on qubit 5 can also be parallelised with the ones in 1 and 4, using Equation 5.23. The optimized measurement pattern is the following:

$$Z_6^{s_1+s_4} Z_3^{s_1} X_6^{s_5+s_1} X_3^{s_1+s_2+s_4} M_2^{\pi/2} X_2^{s_1} M_5^0 M_4^{\theta_4} M_1^{\theta_1} E_G \tag{5.27}$$

Finally, both optimizations can be considered together. In this case, all measurements can be performed at once:

$$Z_6^{s_1+s_4} Z_3^{s_1} X_6^{s_5+s_1} X_3^{s_1+s_2+s_4} M_2^{\pi/2} M_5^0 M_4^{\theta_4} M_1^{\theta_1} E_G \tag{5.28}$$

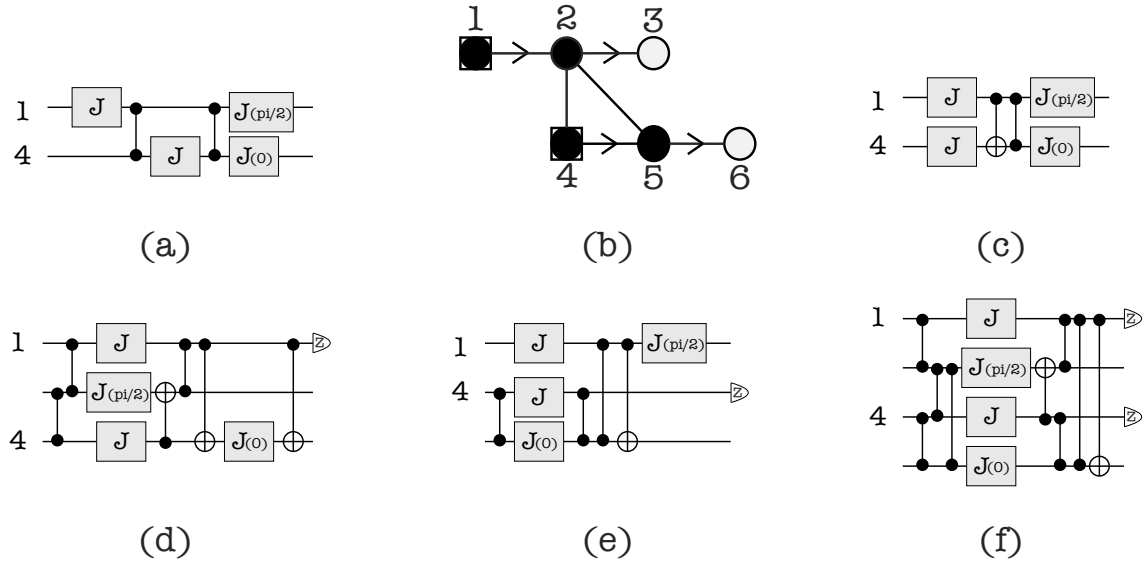


Figure 28: Using compactification procedures to obtain J-gate parallelization for circuits with Pauli angles. See example 2 for more information.

With a few adaptations[6], Algorithm 4 can be applied to the extended circuits associated to the measurement patterns in equations 5.25 to 5.28. The obtained circuits are shown in Figures 28-c to 28-f, respectively. Note that the optimization given by equations 5.22 and 5.23 in the MBQC model becomes a J-gate parallelization in the circuit model. However, it is not clear in which cases the J-gate parallelization implies time optimization. An in-depth analysis of the time-memory tradeoff for extended circuits with Pauli angles remains as an interesting open question.

5.4 Compact circuits from graphs with generalized flow

In this section, I explore some examples of compactification procedures being applied to arbitrary generalized flow patterns. The gflows studied here are different from the ones analyzed in Sec. 5.2 (regflow) and Sec. 5.3 (signal-shifted flow). I start with an introduction (Sec. 5.4.1) giving some intuition on how compactification procedures are designed and then I apply it to a couple of examples in Secs. 5.4.2 and 5.4.3. I conclude this section with a discussion on the applicability of compactification procedures to generic gflows.

[6]Due to the modifications introduced by equations 5.22 and 5.23, it is not possible to create J-blocks for p many wires, where p is the number of times equations 5.22 or 5.23 were used in the original measurement pattern.

5.4.1 Introduction

By definition, arbitrary gflows may require more than one stabilizer operator to correct for each measurement. This results in more than one X correction per measurement (as depicted in Fig. 19) and, consequently, more than one CX gate per $c_{n,i}$ slice. In order to design compactification procedures for arbitrary gflow extended circuits one must identify which CX gate corresponds to the one used in the J-gate identity of Fig. 12, and which CX gates must be removed - it is not obvious how to do this. The identification of this 'special' CX gate is also necessary to implement the procedure of removing CZ gates (as done in Fig. 21), since the procedure reallocates CZ gates to different wires depending on which CX gate we pick to use the identity on. In what follows I will assume this 'special' CX gate has been identified and will be concerned in removing the remaining undesired CX gates, as necessary for the use of the J-gate identity.

In the examples I present in this section, I remove these unwanted CX gates using four of the circuit identities in Fig. 20. First, note that Fig. 20-c has a very similar structure to that of Fig. 20-a, but with CXs instead of CZs, which indicates it may be helpful in removing CXs in a process similar to the one we described above for CZs. However, in order to use this circuit identity, another CX gate is required, and it is not available in the initial entangling round, which consists of CZ gates only. We can make useful CX gates appear using the circuit identity in Fig. 20-b, which transforms a pair of CZ gates from the initial entangling stage into one CZ and one CX gate.

The removal of CX gates will follow basically three steps: (1) transformation of a pair of CZs using the circuit identity in Fig. 20-b, which creates a new CX gate; (2) moving the CX originated in step (1) forward in the circuit using circuit identity 20-a when necessary, and (3) applying the circuit transformation in Fig. 20-c, where the left-most CX is the one generated in step (1), the CX in the middle is the one associated to the J-gate identity and the right-most is the undesired CX, to be removed by this circuit transformation.

To clarify the application of this translation method and to see how more compact circuits are obtained, we now analyze a couple of examples. Note that both examples are of graphs with gflow, for which the star pattern translation method of [33] fails. Moreover, those are examples of graphs with no regflow or SSF and, despite this fact, we obtain compact circuits using our method. The dashed lines in Fig. 29 (first example) and Fig. 30 (second example) identify the set of gates that will be transformed in each step using one of the circuit identities of Fig. 20.

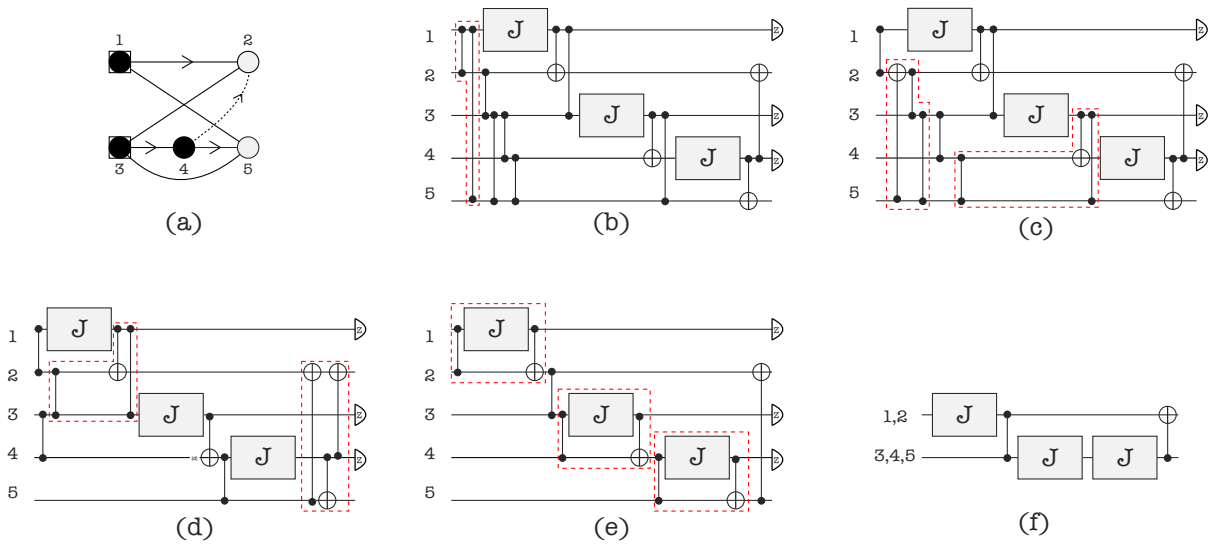


Figure 29: Simplifying an extended circuit originated from the entanglement graph with gflow of Fig. (a). For a step-by-step explanation, see section 5.4.2.

5.4.2 First example

Let us analyze the 1WQC associated with the graph with gflow in Fig. 29-a, where $I = \{1, 3\}$ and $O = \{2, 5\}$. Since the qubit preparation and entanglement structure is already defined by the graph, we must describe the measurements and corrections. Qubits 1, 3 and 4 are measured onto bases $\{|\pm_{\theta_i}\rangle \equiv 1/\sqrt{2}(|0\rangle \pm e^{i\theta_i}|1\rangle)\}$ with respective arbitrary angles θ_1, θ_3 and θ_4. These measurements have correcting sets given by $g(1) = \{K_2\}$, $g(3) = \{K_4\}$ and $g(4) = \{K_2, K_5\}$, with stabilizers $K_2 = X_2Z_1Z_3$, $K_4 = X_4Z_3Z_5$ and $K_5 = X_5Z_1Z_3Z_4$. This information completely characterizes the 1WQC, whose associated extended circuit is shown in Fig. 29-b.

The simplification procedure for this extended circuit goes as follows. In Fig. 29-b we apply the identity from Fig. 20-b, resulting in the circuit in Fig. 29-c; for this circuit we need to apply the identity in Fig. 20-a for the dashed box on the right and the adjoint of the same identity for the dashed box on the left, obtaining thus the circuit in Fig. 29-d. Now we apply the identities in Fig. 20-a and Fig. 20-c to the dashed boxes on the left and right, respectively. After the application of these rewrite rules, we end up with the circuit shown in Fig. 29-e. We can then use the J-gate identity of Fig. 12 to obtain the final compact circuit in Fig. 29-f.

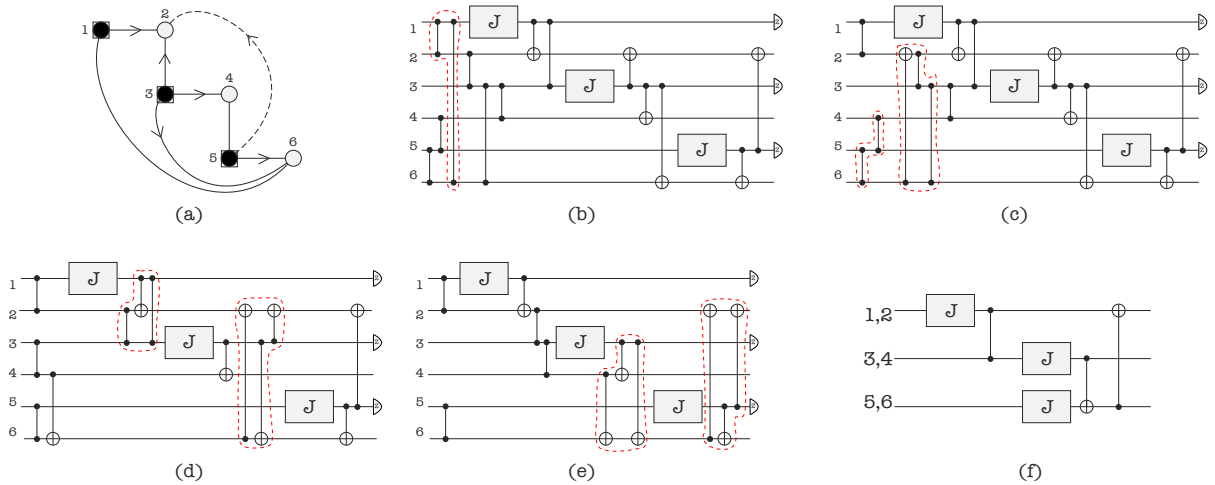


Figure 30: Simplifying the extended circuit originated from the graph in Fig. (a). For a step-by-step explanation, see Sect. 5.4.3.

5.4.3 Second example

Consider the graph in Fig. 30-a, where vertices 1, 3 and 5 represent measured qubits, with respective correcting sets $g(1) = \{K_2\}$, $g(3) = \{K_2, K_4, K_6\}$ and $g(5) = \{K_2, K_6\}$. In Fig. 30-b the extended circuit associated with the graph in Fig. 30-a is shown, where the dashed line encloses the pair of CZs that must be rewritten using the identity of Fig. 20-b. In Fig. 30-c, the dashed line on the left identifies the left-hand side of the circuit identity of Fig. 20-b, which is the rule applied in this case. Also in Fig. 30-c, we use the adjoint of the circuit identity in Fig. 20-a to rewrite the set of gates enclosed by the dashed line on the right. In Fig. 30-d, we again use the circuit identity in Fig. 20-a to rewrite the gates in the left dashed box and use the rule in Fig. 20-c to rewrite the sequence of CX gates inside the right dashed box. The same circuit identity in Fig. 20-c is the one used to reallocate the last two undesired CX gates shown in Fig. 30-e. After the application of these rewrite rules, we end up with the circuit shown in Fig. 30-f, where we have already used the J-gate identity of Fig. 12. Note that this final circuit has only three wires, which is the number of input vertices in the graph of Fig. 30-a.

5.4.4 Discussion

As we pointed out in section 5.4.1, the removal of undesired gates (with the goal of applying the J-gate identity) relies on the existence of certain CZ gates in the initial entangling round, as well as the identification of the 'special' CX gates, the ones to be used in the J-gate identities. For instance, in the second example (Sec. 5.4.3), the open graph under consideration has several different gflows, which means different correction

strategies able to perform the same computation. It turns out that for the gflow function there specified, the special CX gates can be identified in the following way: for every $i, j \in O^C$ find a $k \in T(i)$ and a $l \in T(j)$ (for $T(x) = g(x) \cap N(x)$) such that $k \neq l$. In the aforementioned example we have $T(1) = 2$, $T(3) = 4$ and $T(5) = 6$, which means that the special CX gates are CX_{12}, CX_{34} and CX_{56}. The same identification can be done for the alternate gflow specified by $g(1) = \{4, 6\}$, $g(3) = \{2, 4, 6\}$ and $g(5) = \{2, 4\}$, where the identification $T(1) = 6$, $T(3) = 2$ and $T(5) = 4$ allows the the application of our method, resulting in a compact circuit. On the other hand, the same is not true for the gflow specified by $g(1) = \{4, 6\}$, $g(3) = \{2, 4, 6\}$ and $g(5) = \{2, 6\}$, since $T(1) = T(5)$. It remains as an open question to verify if every graph with gflow has at least one gflow function such that the special CX gates can be identified and a compactification procedure designed, as it was the case for the examples that we analyzed.

5.5 Concluding remarks

In this chapter I have introduced the concept of *compact circuits* and the process of obtaining them from extended circuits, which I have called *compactification procedures*. I have developed two algorithms that implement compactification procedures, one for regflow patterns and the other for SSF patterns. Although the first algorithm gives circuits which are equivalent to the ones given by SPT, the second gives compact circuits for a class of measurement patterns which is out of the scope of application of SPT.

Moreover, Algorithm 4 is able to keep the optimized depth of the SSF extended circuit, since it does not change the structure within the $\mathcal{J}$ layers in the SSF extended circuit. This property of Algorithm 4 will allow us to develop an automated circuit optimization procedure in Chapter 6. In this automated procedure, we translate a circuit to the 1WQC model, optimize the resulting regflow pattern using the signal shifting rules and then translate it back to the circuit model as an SSF extended circuit. Finally, we apply Algorithm 4 to obtain a compact circuit from the SSF extended circuit. Note that such back-and-forth optimization procedure would not be possible if the translation procedure used to bring the computation back to the circuit model as a compact circuit did not preserve the optimization obtained by the application of the signal shifting rules.

6 Optimizing quantum circuits using one-way quantum computation techniques

In this chapter I give an application of the SSF compactification procedure introduced and explored in the last chapter, namely the optimization of quantum circuits. In Sec. 6.1, I describe how a compactification procedure can be used - together with other techniques - to optimize quantum circuits and give a full example of such a procedure. I conclude this chapter with a complexity analysis (Sec. 6.2) of the optimization technique proposed here, comparing it with other known techniques.

6.1 Optimizing quantum circuits by back-and-forth translation

Quantum circuit optimization is a subject of great importance for quantum computing. The enormous technological effort required to perform (even simple) quantum computing tasks makes the research on circuit optimization a potential shortcut to more complex quantum computing tasks, bringing quantum algorithms down to the current technological range. In this chapter I contribute to this discussion by exploring the circuit optimization that can be achieved by combining several different techniques introduced and/or explored in this thesis. More specifically, I reduce the number of $\mathcal{J}$ layers of a circuit, increasing the number of J gates that can be performed in the same computational time slice. As I show in Sec. 6.2, this procedure of reducing the number of $\mathcal{J}$ layers via back-and-forth translation results in the reduction of the total number of computational time slices required to implement a circuit, *i.e.*, the circuit's depth.

The optimization procedure introduced in this chapter is based on the translation of a given quantum circuit [38, 86] into a one-way quantum computation [95, 35]. Naturally, since the two models use different information processing tools, each model has its own

optimization techniques. In the 1WQC model, for instance, most of the optimization techniques are based on the identification of a more efficient correction structure that is directly linked to the geometry of the underlying global entanglement structure (represented as graphs, as we have seen throughout this thesis). Examples of these techniques are the *signal shifting* [35] and the *generalized flow* [24], which were discussed in Secs. 5.3.1 and 3.3.3, respectively. The so-called *standardisation* procedure [35] (discussed in Sec. 3.1.4) can also reduce the number of computational steps by rearranging the 1WQC operations into a normal form. Moreover, *all* Pauli measurements in this model can be performed in the beginning of the computation [96], which is a surprising difference from the quantum circuit model.

On the other hand, most optimization techniques for quantum circuits are based on template identification and substitution. For instance, in [49] some circuit identities are used to modify the teleportation and dense coding protocols, with the purpose of giving a more intuitive understanding of those protocols. Similarly in [103] and [79] a set of circuit identities for reducing the number of gates in the circuit for size optimization was given. In contrast to that, in [84] a useful set of techniques for circuit parallelization was provided, where the number of computational steps is reduced by using additional resources. However, as noted in [103], all the aforementioned circuit optimization techniques are basically exchanging a sequence of gates for a different one without any consideration on the structure of the complete circuit being optimized. The translation into 1WQC would allow us to explore the global structure of a given circuit, by mapping it to a graph and analyzing global properties of it, such as flow conditions. The first such scheme by back and forth translation between the two models was presented in [23]. However the backward translation into the circuit model required the addition of many ancilla qubits. Here I show that the addition of ancilla qubits, in order to keep the optimized number of $\mathcal{J}$ layers, can be avoided in some cases by using the compactification procedure introduced in the last chapter.

The optimization procedure I explore in this chapter, which I will call *signal-shifting circuit optimization (SSCO)*, can be defined in the following way:

Definition 22 ***(Signal-shifting circuit optimization)*** *Let C be a circuit composed by gates taken from the universal gate-set $\{J(\theta), CZ\}$. The application of the signal-shifting circuit optimization (SSCO) can be summarized in the following steps:*

1. *Translate C to the 1WQC model as a regular flow pattern P and associated open graph (G, I, O) using the technique in Def. 16;*

2. *Apply to P the signal-shifting rules (Eqs. 5.8 to 5.11) until all Z_i for $i \in O^C$ have been removed, resulting in the pattern P_{SS};*

3. *Apply the extended translation technique (Def. 14) to the pattern P_{SS}, generating the circuit C_{SS};*

4. *Apply the SSF compactification procedure (Algorithm 4) to the circuit C_{SS}, obtaining the compact circuit C_{Com} as result.*

A few comments regarding the choice of the techniques combined in Def. 22 are worth of note. The first choice, of using star pattern translation to obtain a regflow graph and pattern from a circuit, is obvious: it is the only known technique able to translate circuits to the 1WQC model without getting into the trouble of calculating the full unitary being implemented in one model and then decomposing it appropriately into building blocks of another model. As discussed in Sec. 5.3, the signal shifting is the best optimization that can be applied to patterns with regflow (without changing the underlying open graph) and, therefore, this is the optimization applied in Step 2. To translate the new correcting structure to the circuit model we use the Extended Translation technique, which re-interprets each command in a pattern as a quantum circuit element, giving circuits preserving most of the design of the original pattern: number of qubits, partial order between operations (gates/commands), etc. In the forth step, in order to preserve part of the modified structure, specially the number of $\mathcal{J}$ layers, we use Algorithm 4 to remove all qubits added in Step 3, giving a compact circuit as the final product of SSCO procedure.

In what follows I give a full example of SSCO (Sec. 6.1.1) and then I analyze in terms of complexity the advantages of SSCO, comparing it with different combinations of optimization and translation techniques (Sec. 6.2).

6.1.1 Example

In this example I apply the procedure described in Def. 22 to optimize the circuit in Figure 32-a. The angles of each J-gate are arbitrary and they have been omitted from the figure. The separation into Steps below is a reference to the four steps in Def. 22.

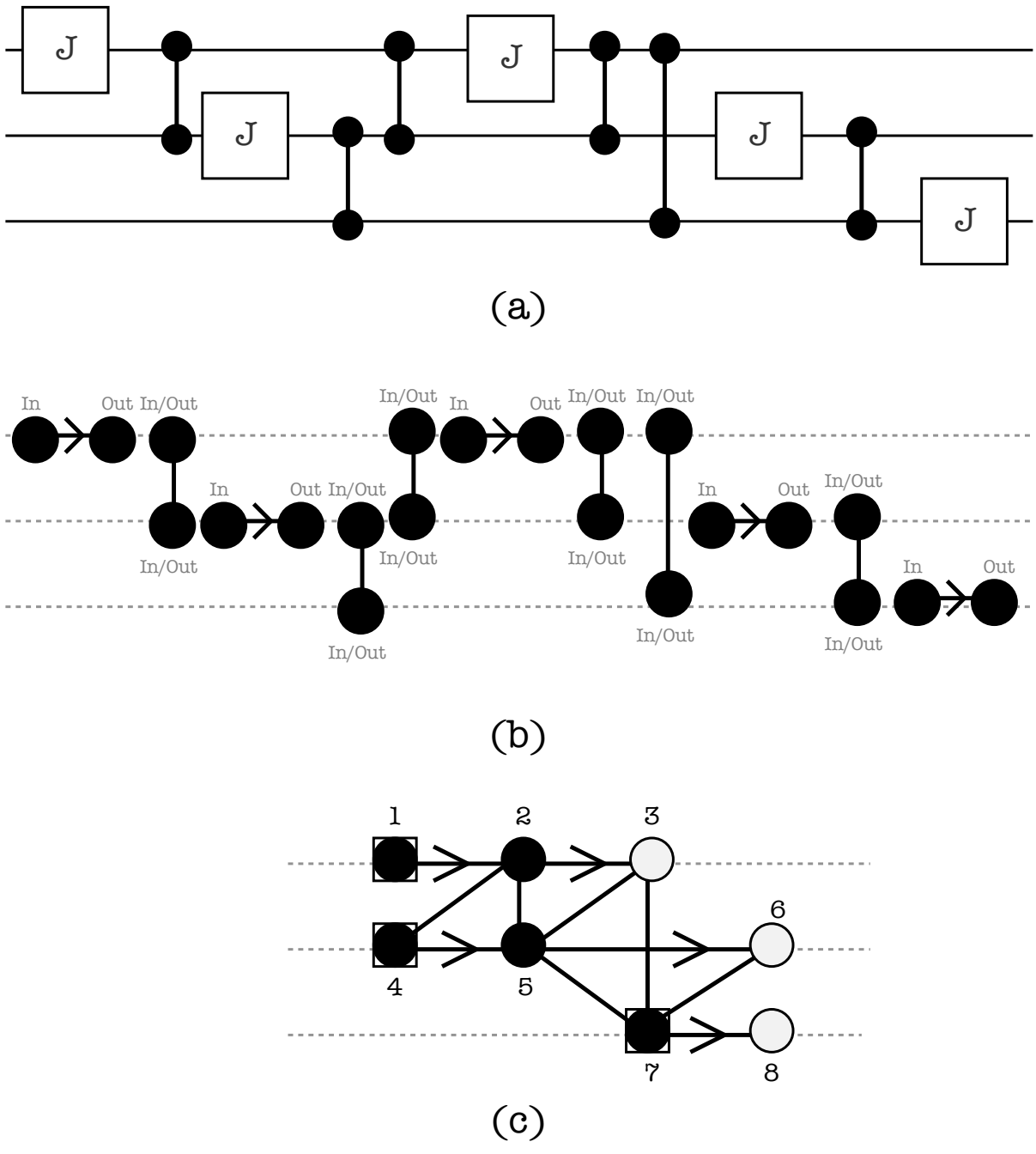


Figure 31: Aplication of the procedure described in Def. 16 to the circuit in Figure 32-a. See Step 1 of Sec. 6.1.1 for more information.

Step 1

Using the method introduced in [23] (summarized in Def. 16 of this thesis), we can translate the circuit in Figure 32-a to the one-way model, resulting in a open graph obeying the regflow conditions (Def. 10) and the associated measurement pattern P. The step-by-step translation to the one-way model is shown in Fig. 31. From the arrows in Fig. 31-b we can infer the regflow function f:

$$f(1) = 2$$
$$f(2) = 3$$
$$f(4) = 5$$
$$f(5) = 6$$
$$f(7) = 8$$

With the open graph and function $f(i)$ defined for all $i \in O^C$, the measurement pattern can be easily constructed:

$$X_8^{s_7} M_7^{\theta_7} Z_7^{s_5} X_6^{s_5} M_5^{\theta_5} Z_7^{s_2} Z_5^{s_2} X_3^{s_2} M_2^{\theta_2} Z_7^{s_4} Z_6^{s_4} Z_3^{s_4} Z_2^{s_4} X_5^{s_4} M_4^{\theta_4} Z_5^{s_1} Z_4^{s_1} Z_3^{s_1} X_2^{s_1} M_1^{\theta_1} E_{78}E_{67}E_{57}E_{56}E_{45}E_{37}E_{35}E_{25}E_{24}E_{23}E_{12} \quad (6.1)$$

which can by more compactly written using the notation in Eq. 3.1 and the abbreviation $E_{1-8} \equiv E_{78}E_{67}E_{57}E_{56}E_{45}E_{37}E_{35}E_{25}E_{24}E_{23}E_{12}$.:

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} Z_3^{s_4+s_1}{}_{s_5+s_4+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4}{}_{s_4}[M_2^{\theta_2}]^{s_1}{}_{s_1}[M_4^{\theta_4}]M_1^{\theta_1} E_{1-8} \quad (6.2)$$

Step 2

The application of signal-shifting to Eq. 6.2 yields:

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} Z_3^{s_4+s_1}{}_{s_5+s_4+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4}{}_{s_4}[M_2^{\theta_2}]^{s_1} \boxed{{}_{s_1}[M_4^{\theta_4}]} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.8)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} Z_3^{s_4+s_1}{}_{s_5+s_4+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4} \boxed{{}_{s_4}[M_2^{\theta_2}]^{s_1} S_4^{s_1}} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} Z_3^{s_4+s_1}{}_{s_5+s_4+s_2}[M_7^{\theta_7}] \boxed{{}_{s_2+s_1}[M_5^{\theta_5}]^{s_4} S_4^{s_1}}{}_{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.10)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} Z_3^{s_4+s_1} S_4^{s_1} \boxed{{}_{s_5+s_4+s_2}[M_7^{\theta_7}] S_4^{s_1}}{}_{s_2+s_1}[M_5^{\theta_5}]^{s_4+s_1}{}_{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4} X_3^{s_2} \boxed{Z_3^{s_4+s_1} S_4^{s_1}}{}_{s_5+s_4+s_1+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4+s_1}{}_{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} X_6^{s_5} \boxed{Z_6^{s_4} S_4^{s_1}} X_3^{s_2} Z_3^{s_4}{}_{s_5+s_4+s_1+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4+s_1}{}_{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} X_3^{s_2} Z_3^{s_4}{}_{s_5+s_4+s_1+s_2}[M_7^{\theta_7}]_{s_2+s_1}[M_5^{\theta_5}]^{s_4+s_1} \boxed{{}_{s_4+s_1}[M_2^{\theta_2}]^{s_1}} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.8)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} X_3^{s_2} Z_3^{s_4}{}_{s_5+s_4+s_1+s_2}[M_7^{\theta_7}] \boxed{{}_{s_2+s_1}[M_5^{\theta_5}]^{s_4+s_1} S_2^{s_4+s_1}} [M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} X_3^{s_2} Z_3^{s_4} \boxed{{}_{s_5+s_4+s_1+s_2}[M_7^{\theta_7}] S_2^{s_4+s_1}}{}_{s_2+s_4}[M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.8)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} \boxed{X_3^{s_2} S_2^{s_4+s_1}} Z_3^{s_4}{}_{s_5+s_2}[M_7^{\theta_7}]_{s_2+s_4}[M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.10)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4}{}_{s_5+s_2}[M_7^{\theta_7}] \boxed{{}_{s_2+s_4}[M_5^{\theta_5}]^{s_4+s_1}} [M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.8)$$

$$X_8^{s_7} X_6^{s_5} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4} \boxed{{}_{s_5+s_2}[M_7^{\theta_7}] S_5^{s_2+s_4}} [M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.11)$$

$$X_8^{s_7} \boxed{X_6^{s_5} S_5^{s_2+s_4}} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4}{}_{s_5+s_4}[M_7^{\theta_7}][M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.10)$$

$$X_8^{s_7} X_6^{s_5+s_2+s_4} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4} \boxed{{}_{s_5+s_4}[M_7^{\theta_7}]} [M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.9)$$

$$\boxed{X_8^{s_7} S_7^{s_5+s_4}} X_6^{s_5+s_2+s_4} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4}[M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_7^{\theta_7} M_4^{\theta_4} M_1^{\theta_1} E_{1-8} \Rightarrow \textit{Eq.}(5.10)$$

$$X_8^{s_7+s_5+s_4} X_6^{s_5+s_2+s_4} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4}[M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_7^{\theta_7} M_4^{\theta_4} M_1^{\theta_1} E_{1-8}$$

Therefore, the SSF pattern P_{SS} is the following:

$$X_8^{s_7+s_5+s_4} X_6^{s_5+s_2+s_4} Z_6^{s_4+s_1} X_3^{s_2+s_4+s_1} Z_3^{s_4}[M_5^{\theta_5}]^{s_4+s_1}[M_2^{\theta_2}]^{s_1} M_7^{\theta_7} M_4^{\theta_4} M_1^{\theta_1} E_{1-8}. \tag{6.3}$$

The difference in depth between the regflow and SSF is shown in the table below:

Method	**depth**	**partial order**
Regflow	5	$1 \prec_f 4 \prec_f 2 \prec_f 5 \prec_f 7$
SSF	2	$1, 4, 7 \prec_s 2, 5$

Step 3

The extended translation (Definition 14) of the measurement pattern in Equation 6.3 gives the circuit in Figure 32-b.

Step 4

Now, Let us apply Algorithm 4 to find the compact form of that extended circuit. We start with the first SSF layer of circuit in Figure 32-b. For $n = 1$ we have $W_1 = \{1, 4, 7\}$, $S^1_{max} = L_f(7) = 5$ and $S^1_{min} = L_f(1) = 1$. Thus, we start with qubit 7. Since $N(s(7))\backslash\{7\} = \{\emptyset\}$, no rewrite procedure is applied. The value of S^1_{max} is decreased to 2 (there is no qubit $i \in W_1$ s.t. $L_f(i) = 3$ or 4). Now we analyse qubit 4 because $L_f(4) = 2$. $N(s(4))\backslash\{4\} = \{2, 3, 5, 6, 7\}$. For qubits in this set it holds that $2 \prec_f 5 \prec_f 7 \prec_f 3, 6$. The maximal value of L_f is stored in $NSTEP$. For both qubits 3 and 6, Algorithm 4 applies RP2, moving E_{65} and E_{35} past CX_{45}.

Now $NSTEP \leftarrow NSTEP - 1$ and the qubit considered is qubit 7. For qubit 7 the algorithm applies RP3, transforming E_{67}, E_{57} and E_{37} into CX_{68}, CX_{58} and CX_{38} respectively and moving those new CXs past the correction structure, removing CX_{48} from the circuit. Now consider qubit 5 and move E_{56} and E_{53} past the correction structure, using RP2. The same rewrite procedure is applied for qubit 2, moving E_{23} and E_{25} past the correction structure. The rewritten circuit is depicted in Figure 32-c.

In the next step, the value of S^1_{max} is decreased to 1, where the qubit to be considered is qubit 1. $N(s(1))\backslash\{1\} = \{2, 3, 4, 5, 6, 7\}$. For qubits in this set it holds that $4 \prec_f 2 \prec_f 5 \prec_f 7 \prec_f 3, 6$. For qubit 6, Proposition 4 is satisfied and hence Algorithm 4 applies RP2, moving E_{65} past CX_{15}. Then for $k = 3$, Algorithm 4 applies RP2 (Proposition 4), moving E_{35} and E_{23} past CX_{12} and CX_{15}. Next, RP4 is applied to $k = 7$, moving CX_{38} and CX_{58} past CX_{13} and CX_{15}. For $k = 5$, we have that $k \in s(i)$ [condition (i) in Proposition 5] and E_{35} in slice $c_{1,1}$ after CX_{15} gate [condition (ii) in Proposition 5]. Therefore, for this case RP2 moves E_{35} and E_{25} past CX_{12} and CX_{13}, creating two CZ_{15} separated by CX_{15}. By Lemma 8, we know those two CZ_{15} can simply be removed from the circuit without changing the computation being implemented. For $k = 2$, Proposition 4 is satisfied and hence RP2 moves E_{25} and E_{23} past CX_{13} and CX_{15}. Finally, for qubit 4, RP3 transforms E_{24} into CX_{25} and move it past CX_{12} and CX_{15}, removing the latter in the process, resulting in Figure 32-d.

In the second SSF layer we have $n = 2$, $W_2 = \{2, 5\}$, $S^2_{max} = L_f(2) = 3$ and $S^2_{min} = L_f(5) = 4$. Thus, we start with qubit 5. Since $N(s(5))\backslash\{5\} = \{7\}$, Algorithm 4 applies RP3, moving CX_{68} past CX_{56} and CX_{58} while removing CX_{58} in the process. After that, the value of $NSTEP$ decreases and qubit 2 is considered. $N(s(2))\backslash\{2\} = \{5, 7\}$ and since $L_f(5) < L_f(7)$, qubit 7 is considered first. In this case Algorithm 4 applies RP4 moving gates CX_{38} and CX_{68} past the correction structure. Now for qubit 5, RP3 is applied

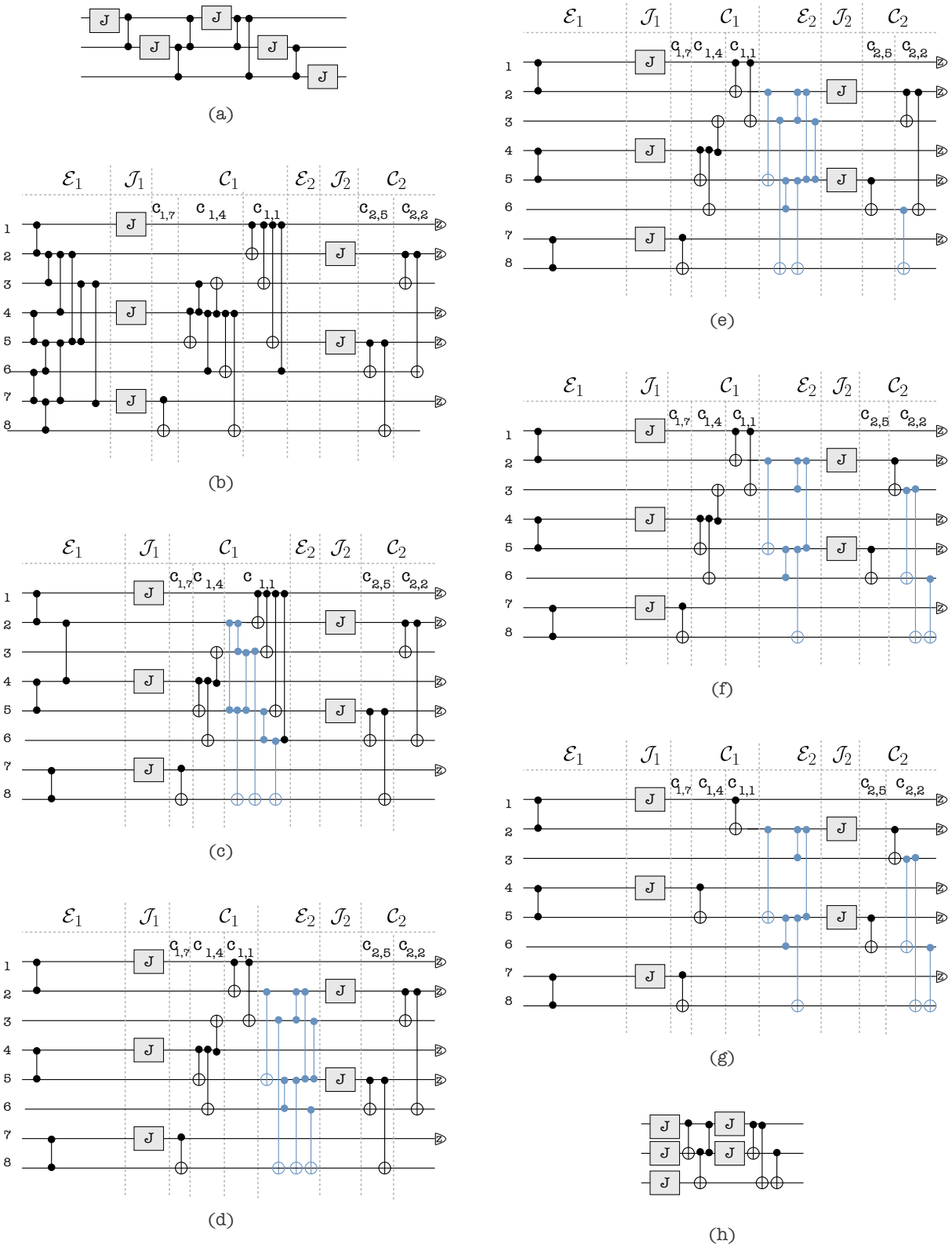


Figure 32: A complete example of global circuit optimization. In each figure the shaded gates are the new gates obtained through the RPs (see Section 6.1.1 for more information). The angles of each J-gate are arbitrary and they have been omitted from the figure.

transforming E_{53} into CX_{36}. The created CX is moved forward past the correction structure, removing CX_{26} from the circuit. The resulting circuit is shown in Figure 32-f. The command in Line 17 of Algorithm 4 removes the remaining undesired gates using the trivial identity $CX_{ij}|+\rangle_j = |+\rangle_j$. The resulting circuit is depicted in Figure 32-g. Finally, the J-gate identity is applied for every qubit $i \in O^C$ resulting in the optimised compact circuit shown in Figure 32-h.

6.2 Complexity analysis and discussion

In this section I analyze the space and depth complexity of the SSCO procedure introduced in this chapter. In special, I show that this method gives circuits less demanding in both time and space, if compared with the general method described in [23]. An overview of this analysis is shown in Figure 33.

Let us start by counting the number of wires and the depth of the computation in the first three steps of SSCO (Def. 22), since they consist in previously analyzed optimization scenarios. Let C be the initial quantum circuit with n wires, $m = \texttt{poly}(n)$ J-gates and depth $d = \texttt{poly}(n)$. We can translate the computation implemented by C to a regflow pattern P with $m+n$ qubits, out of which m are measured qubits, and depth $d_P \leq d$ [23]. This pattern can be further optimized to depth $d_{\text{SS}} \leq d_P$ by performing signal shifting [23] and obtaining a new pattern P_{SS}. The extended quantum circuit C_{SS}, corresponding to the signal shifted pattern P_{SS}, can be created via the method given in Definition 14. This new circuit C_{SS} performs exactly the same computation as the initial circuit C and has the same number of J-gates, $m+n$ wires and depth $d_{\text{SS}} \cdot O(m)$. One could apply the parallelization method of [84], which states that a series of k controlled gates connecting the same input to k target qubits can be parallelized to $O[log(k)]$ depth using $O(k)$ ancillas, to create a parallelized circuit C_{Par} with depth $d_{\text{SS}} \cdot O(\log n)$ and size $O(m^2)$ [23]. Depending on the value of d_{SS}, the depth of C_{Par} can be smaller than that of the original circuit C, but this could increase the space used, which can increase considerably, *i.e* from n to $O(m^2)$.

To resolve this problem, instead of parallelizing the circuit C_{SS} we apply the compactification procedure (Algorithm 4). This will give us the compact circuit C_{Com} (Step 4 in Def. 22) with n wires and $d_{\text{SS}} \cdot O(deg(G))$ where $deg(G)$ is the degree of the open graph (G, I, O) of P_{SS}.

Lemma 12 ***(Lemma 28 in [42])*** *Let C_{SS} be the quantum circuit obtained through*

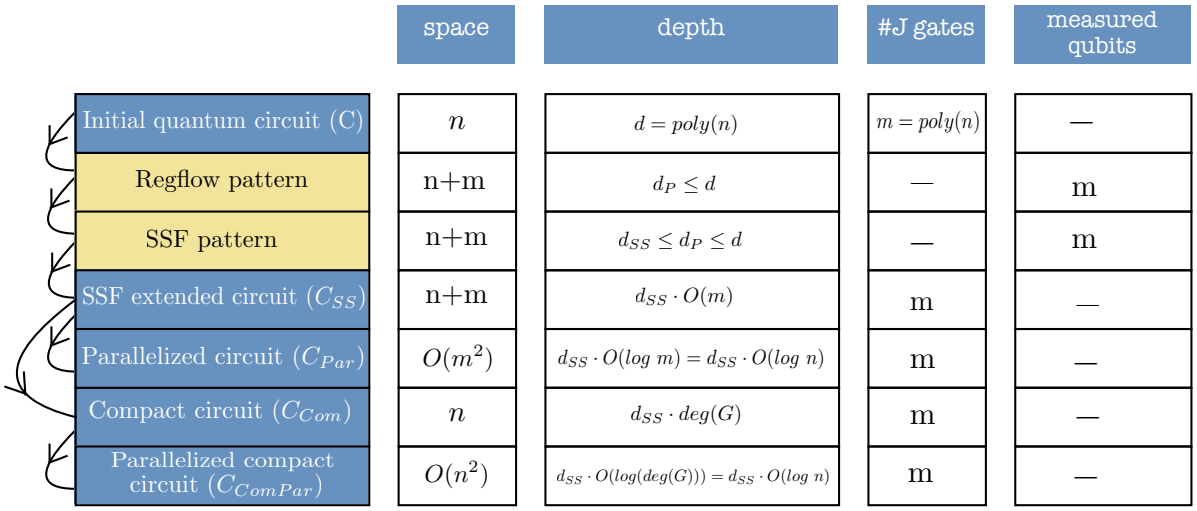


Figure 33: A summary of the optimization procedure using extended translation and compactification. Each arrow on the left of the table indicates a different optimization or translation procedure. The parallelization procedure described in [23] (without Pauli simplification) ends with the "Parallelized Circuit" whereas our optimization can end with a "Compactified Circuit" or a "Parallelized Compactified Circuit" depending the goal of the optimization.

extended translation from a signal shifted measurement pattern P_{SS} on an open graph (G, I, O) with depth d_{SS}. The application of the SSF compactification procedure (Algorithm 4) to C_{SS} results in a circuit with depth $d_{SS} \cdot O(deg(G))$.

Proof. Algorithm 4 applies Rewrite Procedures (2) - (4) (as defined in Section 5.3.4) to the extended circuit of P_{SS}. Note that the number of J-gate layers $\mathcal{J}$ in C_{SS} is equal to the depth of P_{SS} [23]. Since the rewrite procedures do not change the J-gate layers, the compactified circuit C_{Com} will also have d_{SS} J-gate layers.

Now we will count the number of gates in-between the J-gates. We do not need to be concerned with gates originally in the $\mathcal{C}$ layers, since Algorithm 4 removes all of them. As all of the gates in $\mathcal{E}$ layers correspond to edges in the graph G, the maximum number of such gates has to be $O(deg(G))$. According to the rewrite procedures, none of the $CX_{k,i}$ gates created will be moved past the J-gates. As can be seen from the rewrite procedures and Algorithm 4, these will be created because of the existence of some $E_{f^{-1}(i)k}$ gates in C_{SS}. Since these correspond to the edges connected to the vertex $f^{-1}(i)$ there can only exist $O(deg(G))$ many CX gates between any two J-gates. Now we know that there can be a total of $O(deg(G))$ two qubit gates in-between any two J-gates and hence also between the J-gate layers. Therefore the total depth of the circuit C_{Com} will be $d_{\mathrm{SS}} \cdot O(deg(G))$. □

Note that if $deg(G) < \log n$ the C_{Com} will have smaller depth than C_{SS} while the corresponding space will be considerably smaller. We can further decrease the depth of C_{Com} by applying the parallelization method from [84]. The depth of the new circuit C_{ComPar} would be $d_{\text{SS}} \cdot O(\log(degG))$ and size $O(n^2)$. Moreover, due to the way C is translated into P (that is, using the method described in 16), the maximum number of edges connected to a vertex in G will be $O(d)$, where $d = \text{poly}(n)$. Therefore the depth of C_{ComPar} can be written as $d_{\text{SS}} \cdot O(\log n)$. Hence the compactification procedure together with the parallelization method from [84] will in the worst case give us the same depth as the method from [23], but uses considerably less qubits [n^2 vs m^2, where $m = poly(n)$].

To conclude this chapter, I would like to comment on a possible simplification of the procedure in Def. 22. The optimization procedure that I have introduced in this chapter is able to reduce the depth of a circuit by reducing the number of $\mathcal{J}$ layers in it, via back-and-forth translation to the 1WQC model. It is clear, however, that the optimized circuit can be obtained from the original one directly in the circuit model; since both are elements from the same model, there might exist a direct procedure that takes one to another and vice-and-versa. What is not clear is whether a direct procedure within the circuit model would be easier to implement than the one introduced here.

We can learn a couple of things from the example in Sec. 6.1.1. Note, for instance, that the number of two-qubit gates in the original and optimized circuits is preserved, although they might change from CZ to CX gates in some cases. The number of J gates is also preserved, as they are just reallocated in the circuit. I conjecture that the same optimization could be implemented within the circuit model using just the following identity and a prescription of which qubits and in which order the identity must be applied:

$$J_j(\theta)CZ_{ij} = CX_{ij}J_j(\theta). \tag{6.4}$$

This identity can be obtained by noting that $J(\theta) = HP(\theta)$ and $H_jCZ_{ij}H_j = CX_{ij}$. It is easy to check by trial and error that this identity is sufficient to optimize the simple example analyzed in Sec. 6.1.1, but in order to prove that it holds for every circuit optimization obtained by Def. 22, more work should be done.

Therefore, I leave the existence of a simple procedure within the circuit model that is able to implement the same optimization provided by the method in Def. 22 as an open question. It is important to note, however, that independently of there being a simple procedure able to give the same optimized circuits as Def. 22, what I have shown here is that our circuit optimization is the best that can be done by using flow-like techniques,

since all circuits can be mapped to the 1WQC model as graphs with regflow and SSF is the optimal flow for such graphs.

7 Closed timelike curves in one-way quantum computation

The possibility of time travel has been studied for decades in the context of general relativity. Several decades after Einstein developed the general theory of relativity, the first explicit spacetime geometry containing closed timelike curves (CTCs) - paths in spacetime that would allow a physical system to interact with its former self - was proposed by Kurt Gödel [50]. After Gödel's work, a variety of spacetimes containing CTCs were proposed [18, 51]; in fact, it was later realized that CTCs are a generic feature of highly curved, rotating spacetimes [18, 109, 71]. Nevertheless, it is not clear closed timelike curves can in fact exist: Stephen Hawking, for instance, suggested that the conditions necessary to create CTCs could never exist in any physically realizable spacetime - a concept known as Hawking's chronology protection postulate [59]. For instance, it was shown that cosmic string geometries can contain CTCs [51] but cannot create them from scratch [37, 31].

Assuming that closed timelike curves exist, a series of results were obtained regarding their implications for quantum mechanics and quantum computation [39, 12, 28, 15, 92, 93]. In this chapter I describe how the one-way model of measurement-based quantum computation [95] encompasses in a natural way a model for CTCs proposed by Bennett and Schumacher [1], and more recently (and independently) by Svetlichny [106]. I show that the one-way model effectively simulates deterministically a class of CTCs in this model, and characterize this class. A second model for CTCs is Deutsch's highly influential study of quantum time-travel [39]. I show that Deutsch's model leads to predictions conflicting with those of the one-way model, and identify the reason behind this.

This chapter is organized as follows. In Sec. 7.1 I give a brief summary of the recent revival of interest on CTCs by the quantum information community, highlighting the main contributions over the last years. Then, in Sec. 7.2, I review the CTC model based on quantum teleportation and postselection proposed by Bennett, Schumacher and

Svetlichny. Then, in Sec. 7.3 I discuss how CTCs appear naturally in the one-way model, and show that they correspond to CTCs in the Bennett/Schumacher/Svetlichny model. In Sec. 7.4 I characterize a class of CTC-assisted quantum circuits that can be rewritten as time-respecting circuits (*i.e.*, with no CTCs). I give an explicit (although not completely general) method for verifying if a CTC-assisted circuit can be simulated using the 1WQC model. Next, in Sec. 7.5, I review the highly influential CTC model proposed in 1991 by David Deutsch [39]; I use Deutsch's consistency conditions to calculate the predictions for the same CTC-assisted circuits analyzed in Sec. 7.2 using the BSS model. I show that while the BSS model encompasses the one-way model prediction, Deutsch's model gives completely different predictions. Finally, in Sec. 7.6 I discuss a recent proposal by Raussendorf *et al.* [99] for using 1WQC as a quantum mechanical toy model for space-time.

7.1 Review of the literature

In 1991, David Deutsch published his studies on how quantum mechanics would behave in the presence of closed timelike curves (CTCs) [39]. In this seminal paper, Deutsch argues that by treating CTCs, and the particles interacting with it, as quantum mechanical objects, some paradoxical scenarios obtained using classical physics can be avoided. Moreover, it would allow one to "clone" a quantum system (which is prohibited in quantum theory - a result known as the *no-cloning theorem*) and to distinguish between non-orthogonal states. Approximately one decade later, several papers analyzing CTCs from the quantum information and computational complexity perspectives started popping up. In general, those papers address the following question: what is the computational power of a quantum computer if it is allowed to interact with closed timelike curves? In 2002, Todd Brun [27] explicitly showed how CTCs could be used by a classical computer to factor a number into its prime factors; he also argued that the same mechanism could be used to show that classical computers assisted by Deutschian CTCs[1] would be able to solve many hard problems in polynomial time, as `NP`-complete and `PSPACE`-complete problems[2]. In 2003, David Bacon analyzed [12] the computational power of quantum computers assisted by Deutschian CTCs, concluding that they would be able to efficiently solve `NP`-complete problems with only a polynomial number of quantum gates. A few years later, in 2008, S. Aaronson and J. Watrous showed [7] that, if CTCs exist

[1] That is, CTCs modeled as suggested by David Deutsch in [39].

[2] For more complete descriptions of the computational complexity classes that will be used in this thesis, please see: https://complexityzoo.uwaterloo.ca .

and can be described using the Deutsch model, quantum computers would be no more powerful than classical computers. In fact, both types of computers would be extremely powerful, being able to solve PSPACE-complete problems.

In 2009, Brun *et al.* [28] suggested that the assistance of Deutschian CTCs would allow perfect discrimination between any set of quantum states. For instance, they give a CTC-assisted circuit that would allow one to distinguish the four BB84 states $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. In the same year, Bennett *et al.* [15] contested the claims in [28], pointing out that the apparent power of Deutschian CTCs comes from analyzing the evolution of pure states and then extending those results linearly to mixed states. It is not clear, however, that the very concept of mixed states would be well-defined in such CTC-assisted scenarios. More specifically, Bennett *et al.* point out an odd consequence of the results in [28], namely the discrimination of identical states: in [28], they showed how would be possible to map the linear dependent set of states $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$ to four orthogonal states by using the power of Deutschian CTCs, and in [15] the authors point out that it would have as a consequence the ability to discriminate the identical mixed states $I/2 = \frac{1}{2}\big(|0\rangle\langle 0| + |1\rangle\langle 1|\big)$ and $I/2 = \frac{1}{2}\big(|+\rangle\langle +| + |-\rangle\langle -|\big)$. An active discussion on the ability of CTC-assisted quantum circuits to distinguish non-orthogonal states, clone quantum states and to perform other post-quantum-theory tasks has been taking place since then [29, 30, 93, 70, 102, 9, 111].

In 2009, George Svetlichny suggested a new way of analyzing CTC-assisted quantum circuits [106]: instead of requiring Deutsch consistency conditions, he proposes the use of an adaptation of the quantum teleportation protocol where one is allowed to post-select measurement outcomes. In fact, this model had been independently proposed by Bennett and Schumacher in an unpublished joint work [1] presented at the quantum information processing school *QPIP* (Mumbai, India), in February 2002. Therefore, from this point on I will refer to this CTC model as the BSS model, after Bennett, Schumacher and Svetlichny.

The BSS model leads to less powerful CTC-assisted circuits than the Deutsch model does. A BSS circuit would be able to solve problems in the complexity class post-BQP (which is believed to be larger that NP but properly contained in PSPACE), and since it was proved in [5] that post-BQP = PP, we say that BSS circuits are able to solve PP-complete problems. Therefore, BSS CTCs would be less powerful than Deutschian CTCs, which apparently can solve PSPACE-complete problems.

In 2010, in collaboration with Ernesto F. Galvão and Elham Kashefi, I proposed a

framework to analyze CTCs in the 1WQC model [41]. We showed that 1WQC algorithms can be formally manipulated (using the stabilizer formalism) in order to allow the appearance of anachronical operators, which become anachronical gates when translated to the circuit model. Those circuits with anachronical gates can be further manipulated (simply by adding a $SWAP$ gate and re-drawing the circuit) to become CTC-assisted circuits. We also showed that those CTC-assisted circuits originated from anachronical 1WQC algorithms encompass the results given by the BSS model, while showing a general disagreement with the predictions of the model proposed by Deutsch. This chapter is mainly based in the results of that paper.

In the same year an experimental simulation [76] and a deeper theoretical analysis [75] of the BSS model was uploaded to the preprint repository Arxiv.org by Lloyd *et al.*, being published a few months later. Those papers reinforced our result that the BSS model seems to be more adequate to describe CTC-assisted circuits than the model proposed by Deutsch.

The research on the power of CTCs seems to be far from an end, since there is still no agreement on which of the existing CTC models (BSS and Deutsch's) should be used to describe CTC-assisted circuits. In fact, there is still no good reason to choose one of those two models and maybe some new, interesting ways of analyzing CTC-assisted circuits may be proposed in the future.

Remark on terminology. Contrary to most of the papers that analyze CTCs using the model proposed by C. Bennett and B. Schumacher [1] and independently by G. Svetlichny [106], I use the expressions "BSS CTCs" and "BSS model" (in reference to Bennett/Schumacher/Svetlichny - BSS) as opposed to "P-CTCs" and "P-CTC model" (in reference to postselected CTCs) in order to give due credit to the authors of the model, as is done with the Deutsch model for CTCs [39].

7.2 A model for CTCs based on teleportation and post-selection: The BSS model

In this section I review the main features of the CTC model proposed by Bennett and Schumacher [1] and by Svetlichny [106], the BSS model. This model combines quantum teleportation with post-selection in an attempt to predict the behavior of quantum mechanics near closed timel-like curves. Since this model relies on post-selection, anything that happens in a spacetime with BSS CTCs can also happen in conventional quantum

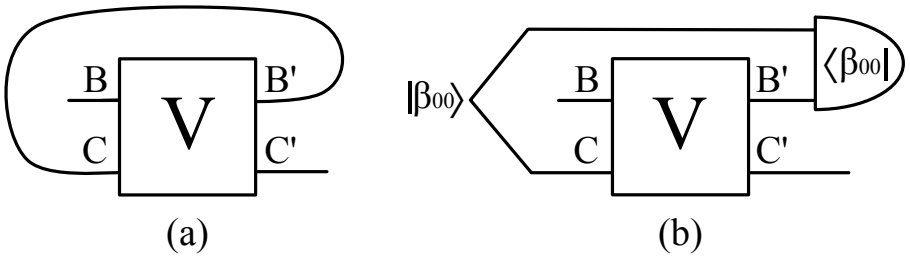


Figure 34: a) CTC takes qubit back in time to interact with its past self; b) Bennett/Schumacher/Svetlichny (BSS) circuit to simulate this CTC probabilistically, using teleportation and post-selection (see main text).

mechanics with some finite probability.

Ideas similar to the BSS model were proposed independently in several different contexts. For instance, the combination of entanglement and deterministic projections (instead of quantum measurements) were studied in the context of black hole evaporation by Horowitz and Maldacena [61] and Gottesman and Preskill later noted that this scheme could be used for time travel [53]. Pegg explored a similar mechanism for probabilistic time machines [90]. Chiribella *et al.* also used this mechanism when studying extensions of the quantum computational model [32] and Brukner *et al.* when analyzing probabilistic teleportation, where only desirable results were retained, as a computational resource [26].

More recently in [75], Lloyd *et al.* discussed several aspects of the BSS model, as for instance its consistency with path integral approaches for spacetimes containing closed timelike curves. The first experimental implementation of the BSS model was published in [76], where the authors implemented a simulation of the grandfather paradox.

For simplicity, I will restrict our discussion to two-qubit unitaries; the generalization to larger-dimensional systems is straightforward. In Fig. 34-a I represent a circuit with a CTC that takes the top qubit back in time to interact with its past self via the two-qubit unitary V. This CTC is simulated using teleportation in BSS's construction (see Fig. 34-b). Two qubits are prepared in the Bell state $|\beta_{00}\rangle = 1/\sqrt{2}(|00\rangle + |11\rangle)$, with one of them sent though V together with an arbitrary input state $|\psi_{in}\rangle$ at position B in Fig. 34-b. After V we perform a Bell-state measurement, postselecting those events corresponding to projection onto the initial $|\beta_{00}\rangle$. The post-selected teleportation guarantees that the state at C is state B' teleported back in time to interact with state B via V. The scheme works only probabilistically, implementing a map from state $B \to C'$.

In the absence of yet-undiscovered physical CTCs, quantum circuits such as the BSS

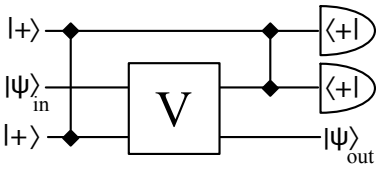


Figure 35: Rewritten BSS circuit using preparation of and projections onto $|+\rangle$ states. The unitary V is decomposed using the universal gate-set consisting of $J(\theta)$ and CZ.

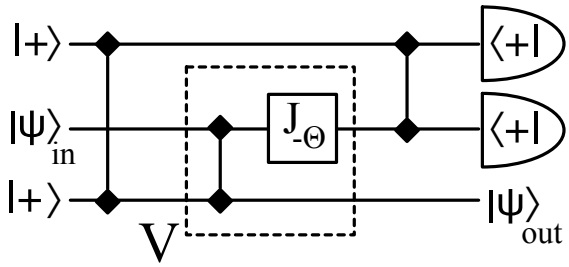


Figure 36: BSS circuit simulating a CTC with unitary $V = [J(-\theta) \otimes I]CZ$.

circuit in Fig. 34-b simulate the CTCs with a finite probability of success. Svetlichny's model differs from that of Bennett and Schumacher in an irrelevant detail only: the unitaries he considered involved a swap between the states to be fed to V, that is, he modelled CTCs such as the one in Fig. 34-a with unitaries of the form $V = U \cdot SWAP$.

In order to link the BSS CTC simulation circuits with the one-way model of quantum computation [95] we consider the universal gate set $\{J(\theta), CZ\}$ [34]. We can rewrite any BSS circuit using these gates, initialization in state $|+\rangle \equiv 1/\sqrt{2}(|0\rangle + |1\rangle)$, and final Pauli X measurements that post-select projections onto state $|+\rangle$ (see Fig. 35). This is closely related to the setting of the one-way model, so we can use various tools developed in that context to study CTCs as described by the BSS model.

Let us work out the BSS simulation for one particular CTC of interest, given by unitary $V = [J(-\theta) \otimes I]CZ$. The BSS circuit that simulates this particular CTC is shown in Fig. 36. The circuit acts on input state $|\psi_{in}\rangle = \alpha|0\rangle + \beta|1\rangle$ to output state $|+\rangle$ with probability $|\alpha + e^{-i\theta}\beta|^2/4$, as can be easily checked. The nonzero probability of projecting onto $|+\rangle$ state means the BSS formalism predicts that the action of this CTC is to deterministically project the input state onto $|+\rangle$.

Interestingly, the finite probability of success is the mechanism that avoids paradoxical scenarios to occur, *i.e.*, situations in which the combination of input state and interaction V prevents the existence of a self-consistent state for the time-traveling system. In such scenarios, the BSS model yields a probability of success equal to zero, as noted in [1, 75].

This mechanism encompasses in a natural way the Novikov principle [48], which states that only logically self-consistent events can occur in the Universe. Note also that any quantum measurement yields the same statistical results (and here I include correlations with time-respecting qubits) independently of whether it is performed in the system entering the BSS CTC or exiting it, since the BSS simulation is constructed by projecting out part of a pure state. Contrary to Deutschian CTCs, which typically take pure states to mixed states, BSS CTCs take pure states to pure states. Therefore, BSS CTCs behave like an ideal quantum channel, taking quantum states to the past [58].

7.3 Closed timelike curves in one-way quantum computation

In this section I show how CTCs appear naturally in the one-way model of quantum computation [96]. The key element is the appearance of anachronical dependencies in measurement patterns, obtained using the stabilizer formalism. This property was explored in depth in Chapter 3, where it was shown that deterministic measurement patterns [associated to an open graph (G, I, O)] can be constructed in two steps: (i) start with a projection-based pattern, where every qubit $i \in O^C$ is projected onto state $|+_{\theta_i}\rangle$ [Eq. (3.40)] and (ii) look for a collection of stabilizer operators K_j ($j \in I^C$) that removes every anachronical Z from the pattern without creating new ones. In the cases step (ii) can be successfully implemented, the new pattern is a physically sound measurement pattern, with no projective operations. Here we study these problematic time dependencies in more detail, and show that they correspond to BSS CTCs when translated to the circuit model.

For concreteness, let us start by analyzing a simple pattern implementing a one-qubit unitary:

$$X_2^{s_1} M_1^{\theta} CZ_{12} N_2 |\psi_{in}\rangle_1. \tag{7.1}$$

This measurement pattern was analyzed before in this thesis in Chapters 3 and 5, but for the sake of convenience let us review again what it is doing. This sequence of operations can be represented as a two-qubit quantum circuit, see Fig. 37-a. An arbitrary input state $|\psi_{in}\rangle_1$, previously entangled via a CZ gate with a qubit initially in state $|+\rangle_2$, is then measured in the $|\pm_\theta\rangle$ basis. The outcome $s_1 = 0$ or 1 controls classically whether or not to apply a Pauli X gate on qubit 2. Fig. 37-b represents the same operations, only with a Z basis measurement and with the controlled operation implemented coherently

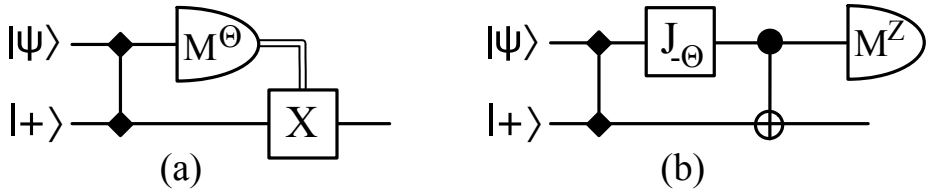


Figure 37: Two equivalent circuits, i.e. implementing the same unitary. In a) we have a classically-controlled X unitary dependent on the measurement outcome of $|\pm_\theta\rangle$ basis projection. In b) this has been turned into a coherent circuit with measurement onto the Z basis.

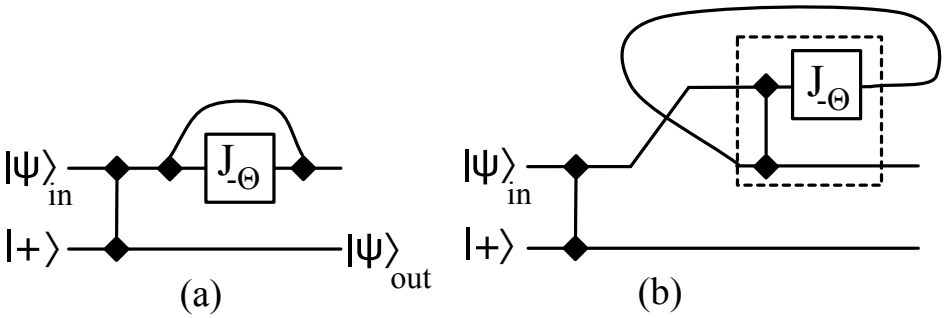


Figure 38: a) Circuit that includes a CTC with an anachronical CZ gate; it is equivalent to the two circuits in Fig. 37. b) The same circuit rewritten in the BSS format.

as a CX gate. The two circuits are equivalent as they implement the same unitary $J(-\theta)$ on initial state $|\psi_{in}\rangle_1$, with output in qubit 2.

It is easy to find other patterns (and corresponding circuits) which are equivalent to sequence (7.1), *i.e.*, implement the same unitary $J(-\theta)$ between input and output qubits. We start by observing that the state to be measured $|G\rangle = CZ_{12}|\psi_{in}\rangle_1|+\rangle_2$ is stabilized by the two operators $\{1 = Z_1^0 X_2^0, Z_1^1 X_2^1\}$, as $Z_1 X_2 |G\rangle = |G\rangle$. In other words, $Z_1^{s_1} X_2^{s_1}$ is a stabilizer of $|G\rangle$ independently of whether $s_1 = 0$ or 1. This enables us to manipulate sequence (7.1) as follows:

$$X_2^{s_1} M_1^\theta |G\rangle = X_2^{s_1} M_1^\theta Z_1^{s_1} X_2^{s_1} |G\rangle = M_1^\theta Z_1^{s_1} |G\rangle. \tag{7.2}$$

This last sequence represents a time-travel conundrum: a classically controlled Pauli Z unitary which must be applied depending on the outcome of an as-yet unmeasured qubit. This is turned into a quantum CTC if we apply the anachronical Pauli Z operation coherently, as we see in Fig. 38-a. A rewriting of this circuit in slightly different form (Fig. 38-b) shows that the top qubit enters exactly the CTC we analyzed using the BSS model (see Fig. 36). We can now compare the predictions of the one-way model for this particular CTC with those given by the BSS model.

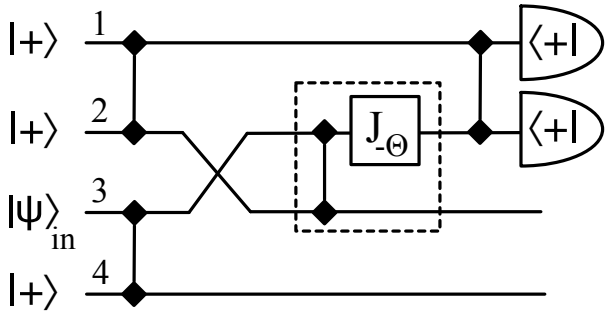


Figure 39: Circuit that implements the probabilistic BSS simulation of the CTC circuit in Fig. 38-b.

An apparent mismatch between the BSS formalism and the one-way model appears when we analyze the action of the CTC in Fig. 38-b. On the one hand, our analysis of the circuit in Fig. 36 has shown that the CTCs effect is to project the input state onto $|+\rangle$. On the other hand, comparison between Figs. 37-a and 38-a suggests that the effect of the CTC should be to project the top qubit onto $|+_\theta\rangle$ instead. This is because a post-selected $|+_\theta\rangle_1$ projection in the circuit of Fig. 37-a is what it takes to implement the unitary $J(-\theta)$ to $|\psi_{in}\rangle$, without the need for the controlled X correction.

The resolution of this apparent conflict is surprising. The one-way model only predicts that, when embedded in the circuit of Fig. 38-a, the CTC should implement the same input-output map $J(-\theta)$ as its two equivalent circuits in Fig. 37. Using the circuit in Fig. 36 to simulate what happens in the CTC of Fig. 38 is unwarranted; instead, we should simulate the CTC's action when embedded in the circuit of Fig. 38. The BSS circuit for this simulation is in Fig. 39. A simple calculation shows that this circuit, at once, fulfills the predictions of both the BSS and the one-way model: it projects qubits 3 and 4 onto state $|+\rangle_3 \otimes [J(-\theta)|\psi_{in}\rangle]_4$.

This illustrates what seems to be a general feature of CTCs: their effect extends not only to the time-travelling sub-system A, but to all sub-systems that have interacted with A prior to A's encounter with CTCs. In the context of the one-way model what interests us is the dynamics the measured (time-travelling) qubits induce on the output (time-respecting) qubits. It is this dynamical map that we can calculate and compare, as we have done in this section. This will also be the key that allows for the comparison with Deutsch's CTC model in Sec. 7.5.

7.3.1 Obtaining CTC-assisted circuits from one-way patterns

The simple form of the one-way pattern in Eq. (7.1) may suggest that the CTCs that appear in the one-way model are only as simple as the one appearing in Fig. 38, with CTC unitary V consisting of only two gates, a CZ and a $J(\theta)$. In fact, a given deterministic one-way computation can be equivalent to the simulation of different CTCs implementing the same input-output map, and these CTCs may have different structures. To illustrate this, let us consider the following sequence of commands implementing a one-way computation:

$$X_2^{s_4} X_1^{s_4} Z_1^{s_3} M_4^{\theta_4} X_4^{s_3} M_3^{\theta_3} |G\rangle, \tag{7.3}$$

where

$$|G\rangle = CZ_{23} CZ_{13} CZ_{14} CZ_{34} N_4 N_2 N_1 |\psi_{in}\rangle_3 \tag{7.4}$$

is the state associated with the graph in Fig. 40-a. The extended translation of the measurement pattern in Eq. (7.3) is shown in Fig. 40-b. Note that as the sequence in (7.3) is time-respecting, so is the associated circuit.

We can now obtain different sequences of one-way operations that implement the same map, by rewriting the initial state $|G\rangle$ as $K_i|G\rangle$, with K_i being a stabilizer of $|G\rangle$. It is easy to check that the following operators are stabilizers of $|G\rangle$:

$$K_1^{s_4} = X_1^{s_4} Z_3^{s_4} Z_4^{s_4} \tag{7.5}$$

$$K_2^{s_4} = X_2^{s_4} Z_3^{s_4} \tag{7.6}$$

$$K_4^{s_3} = X_4^{s_3} Z_3^{s_3} Z_1^{s_3} \tag{7.7}$$

Note that as qubit 3 is in an arbitrary input state, $K_3 = X_3 Z_1 Z_2 Z_4$ is not a stabilizer of $|G\rangle$. Using stabilizers K_1, K_2, K_4 we obtain three new sequences of operations that now include anachronical corrections. For example, after applying $K_2^{s_4}$ we have the following pattern:

$$X_1^{s_4} Z_1^{s_3} M_4^{\theta_4} X_4^{s_3} M_3^{\theta_3} Z_3^{s_4} |G\rangle, \tag{7.8}$$

which has the anachronical command $Z_3^{s_4}$. A different anachronical pattern can be obtained by using the identity $K_4^{s_3}|G\rangle = |G\rangle$ in Eq. (7.3):

$$X_2^{s_4} X_1^{s_4} M_4^{\theta_4} M_3^{\theta_3} Z_3^{s_3} |G\rangle, \tag{7.9}$$

which has the anachronical command $Z_3^{s_3}$. Again, we can obtain a new anachronical

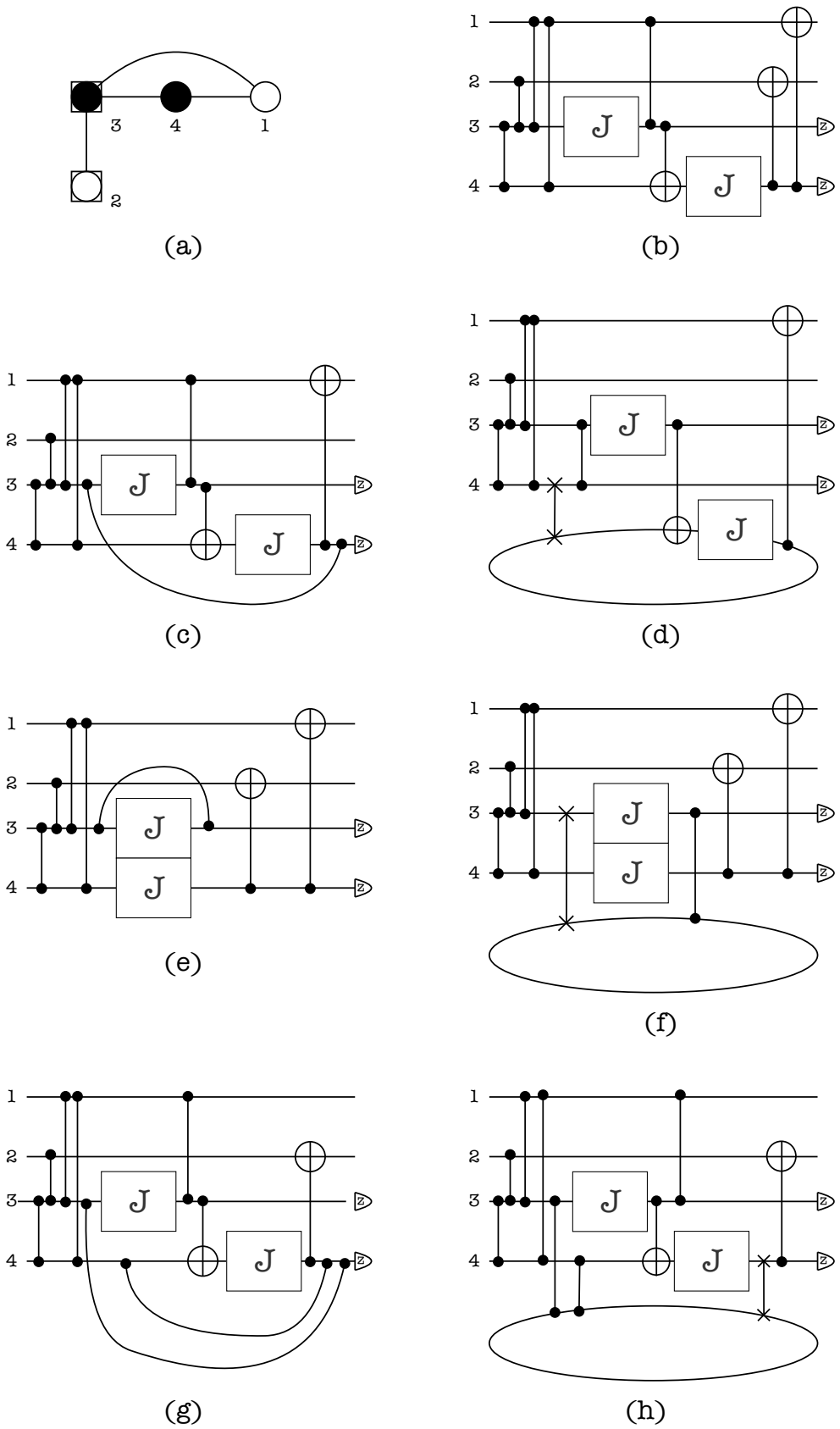


Figure 40: (a) Entanglement graph corresponding to state $|G\rangle$ in Eq. (7.4). On this state we can perform the sequence of one-way operations in Eq. (7.3). The extended translation of Eqs. (7.8), (7.9) and (7.10) are shown in (c), (e) and (g), respectively. In Figs. (d), (f) and (h) I redraw the CTC circuits in Figs. (c), (e) and (g), respectively, so as to explicitly show the CTC as a time-travelling qubit.

pattern by using the identity $K_1^{s_4}|G\rangle = |G\rangle$ in Eq. (7.3):

$$X_2^{s_4} Z_1^{s_3} M_4^{\theta_4} Z_4^{s_4} X_4^{s_3} M_3^{\theta_3} Z_3^{s_4} |G\rangle, \tag{7.10}$$

which has the anachronical commands $Z_3^{s_4}$ and $Z_4^{s_4}$.

Each such sequence can be translated into the circuit model, where the classically controlled X and Z corrections appear as coherent CX and CZ gates. These anachronical gates correspond to CTCs when translated to the circuit model. The extended translation of Eqs. (7.8), (7.9) and (7.10) are shown in Figs. 40-c, 40-e and 40-g, respectively. In Figs. 40-d, 40-f and 40-h I redraw the CTC circuits in Figs. 40-c, 40-e and 40-g, respectively, so as to explicitly show the CTC as a time-traveler qubit.

Any deterministic one-way pattern yields a class of CTC circuits simulatable by it. These circuits are obtained as shown above, by using arbitrary stabilizers that introduce the anachronical dependencies in the (originally deterministic and time-respecting) pattern.

7.4 Deterministic simulations of CTCs

Note that a deterministic simulation of the BSS circuit in Fig. 39 (with respect to its action on input state $|\psi_{in}\rangle$)) is achieved by the circuit in Fig. 37-b, which effectively implements unitary $J(-\theta)$. In other words, the circuit in Fig. 37-b simulates deterministically the CTC circuit in Fig. 38-b. A natural question is then to determine which BSS CTCs can be simulated deterministically by a one-way pattern and its equivalent circuit. In this section I present a systematic way to find measurement patterns that deterministically simulate CTCs in BSS model.

We start with the BSS circuit in which each CTC is simulated by preparation of state $|\beta_{00}\rangle$ and subsequent postselected projection onto the same state. The first step is to translate the BSS circuit into a one-way projection-based pattern[3]. We translate the BSS circuits using a simple variation of the star pattern translation technique (introduced in [23] and summarized in Def. 16, Chapter 4). This variation consists in the modification of Steps 1 and 3 and the addition of another Step in Def. 16. First, the technique in Def. 16 does not account for postselection, necessary to translate the post-selected Bell-pair measurements in the BSS circuit. A postselected projection is translated to the one-way

[3]That is, a pattern with projections $P_i^{|\theta_i\rangle}$ instead of measurements $M_i^{\theta_i}$ and time-respecting Pauli corrections, as in Eq. (3.41).

model as expected, that is, as a vertex i being acted upon by the operator $P_i^{|+\rangle} = M_i^0 Z_i^{s_i}$ (a post-selected Pauli X measurement). Moreover, a new subcase shall be added to Step 3 in Def. 16 in order to take into account the new label that refers to postselected measurements. Next, we need to modify Step 1 in Def. 16 in order to give a projection-based pattern implementing the $J(\theta)$ gate. We do this translation by replacing each J gate in the CTC-assisted circuit by a horizontal edge connecting to vertices. Label the vertex on the left as "input" and the one in the right as "output". In the associated pattern, the qubit in the left is projected onto state $|+_\theta\rangle$. These modifications are summarized in Def. 23, where I give a complete description of the method that allows one to find time-respecting circuits implementing the same map as some CTC-assisted circuits.

The resulting pattern implements the same input-output map as the original BSS circuit and it can be manipulated using stabilizer operations (such as local complementation [107]), with the aim of eliminating the ancillas added in the $|\beta_{00}\rangle$ state preparations and postselections required by the BSS simulation circuit. This can always be done since any such Bell-pair projections translate only as a sequence of Pauli projections, which enables us to apply the general rules for removing a Pauli measurement from a measurement pattern [60]. In some cases this results in a pattern where the anachronical corrections (added during the translation of the J gate) can no longer be eliminated, resulting in anachronical circuits corresponding to unsound physical operations.

The next step is to turn the projection-based pattern into a runnable measurement pattern (Def. 5). In some cases it will not be possible to do so, which is reasonable since CTC-assisted quantum circuits are believed to be more powerful that regular quantum circuits, as will be discussed in Sec. 7.5.3. In other cases, however, the open graph associated to the resulting pattern satisfies the determinism conditions for the one-way model (the flow conditions) discussed extensively in Chapter 3. If that is the case, the anachronical corrections that appear can be removed. Note that the operations used in removing the ancillas introduced by the BSS simulations consist of Local Complementation, removal of Pauli measurements, and other stabilizer manipulations, all of which preserve the map implemented from input to output. As a result, the newly-found sequence (and its equivalent circuit) implements deterministically the same map that succeeded only probabilistically in the BSS simulation circuit. This effectively characterizes a class of BSS CTC circuits that admit a deterministic simulation in the one-way model. The method of obtaining a time-respecting circuit from a CTC-assisted circuit via one-way model (when it is possible) is summarized in Def. 23.

Definition 23 ***(Obtaining time-respecting circuits from CTC-assisted ones)*** *Let C_{ctc} be a CTC-assisted circuit composed by gates from the gate-set $\{J(\theta), CZ\}$. Then, a time-respecting circuit performing the same input to output map can be obtained if all steps below can be accomplished succesfully:*

1. *Apply the method for obtaining patterns from circuits described in Def. 16 with the following modifications:*

   - *(Step 0) A postselected projection is translated as a vertex i with label "post-s" being acted upon by the operator $P_i^{|+\rangle} = M_i^0 Z_i^{s_i}$ (a post-selected Pauli X measurement);*
   - *(Step 1 modified) Replace each J-gate by a horizontal edge connecting two vertices. Label the vertex on the left as "input" and the one in the right as "output". In the associated pattern, the projective operator $P^{|+_\theta\rangle}$ is applied to the qubits on the left, but no operator is applied to the qubit on the right;*
   - *(New Step 3 subcase) A vertex labeled "post-s" is contracted with any other vertex label as one vertex with "post-s" label;*

   As a result, we obtain a projective pattern and associated open graph.

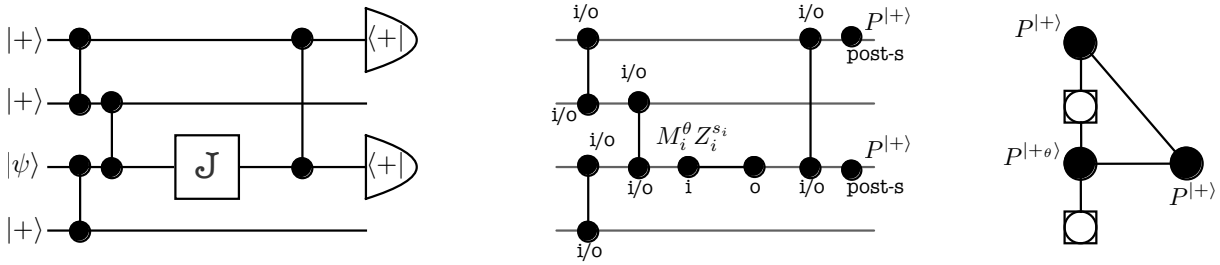
2. *Apply local complementation [107] to turn postselected X measurements into postselected Z measurements. Remove from the graph the vertices being acted upon by a postselected Z measurements (see Sec. 2.5.2).*

3. *Verify if the open graph obtained in Step 2 satisfies the generalized flow condition. If it does, we obtain a runnable measurement pattern.*

4. *Translate the measurement pattern obtained in Step 3 to the circuit model using the extended translation technique (Def. 14).*

Note that if the open graph obtained in Step 2 satisfies the regular flow or the signal-shifted flow conditions, more economical time-respecting circuits can be obtained by applying the compactification procedure in Algorithms 1 or 4, respectively, to the corresponding extended circuit.

As an example, I work out explicitly the stabilizer manipulations required to turn a (probabilistic) BSS CTC simulation circuit into a deterministic one. This will be done for the BSS circuit in Fig. 39.



(a) (b) (c)

Figure 41: Translating the BSS circuit in Fig. 39 to the one-way model (see Sec. 7.4.1).

7.4.1 Example

Let us now translate the circuit in Fig. 39 into a one-way measurement pattern using the techniques described in Def. 23. In Fig. 41 we show the application of Step 1 in Def. 23. As a result, we obtain a graph in Fig. 41-c and the associated projection-based pattern:

$$P_1^{|+\rangle} P_5^{|+\rangle} P_3^{|\theta\rangle} |G\rangle, \tag{7.11}$$

$$\begin{aligned} |G\rangle \equiv\ & CZ_{34}CZ_{35}CZ_{32}CZ_{51}CZ_{12} \\ & N_1 N_2 N_4 N_5 |\psi_{in}\rangle_3, \end{aligned} \tag{7.12}$$

where the qubits' sub-indices match those of the input qubits in the circuit of Fig. 39, except qubit 5 which is a new qubit added in the pattern so as to implement the J gate. Qubit 3 is our input state, and all the others are prepared in state $|+\rangle_j$ by the N_j command. The graph corresponding to state $|G\rangle$ in Eq. (7.12) is shown in Fig. 42-a. The projective X operators in Eq. (7.11) can be dealt with by using the so-called *local complementation* rules (Step 2 in Def. 23) introduced in [107], which we now review.

Local complementation is an operation that changes a state in a way that is most conveniently described by the change in its stabilizers. It corresponds to local unitaries applied on a chosen qubit and its neighbors in the graph, and has been shown to preserve the computation that can be performed using the state in the one-way model [60, 107, 97]. First, let us recall the definition of the phase gate $S = |0\rangle\langle 0| + i|1\rangle\langle 1|$ and define the unitary $C = HSH = 1/\sqrt{2}(e^{\frac{i\pi}{4}}1 + e^{\frac{-i\pi}{4}}X)$, where H is the one-qubit Hadamard gate. In

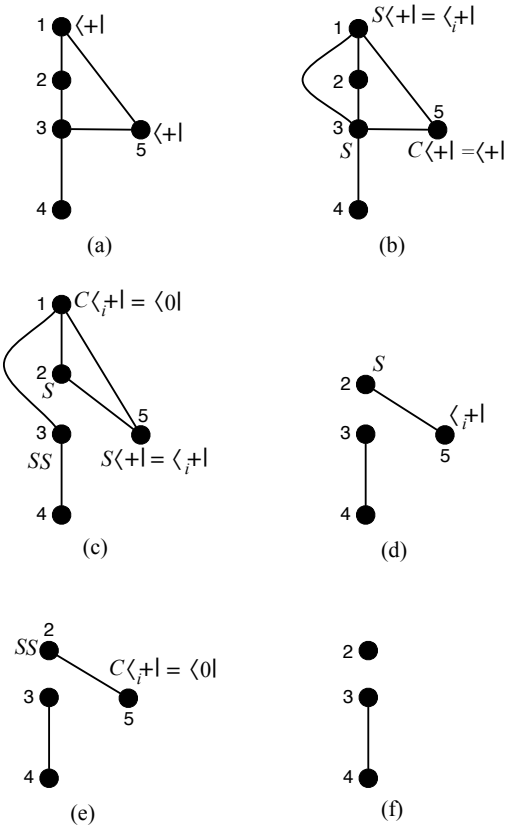


Figure 42: Stabilizer manipulations that simplify the one-way sequence (7.11); $\langle_i+|$ denotes a projection onto $|0\rangle + i|1\rangle$. a) Graph representing initial entanglement structure and operations. Qubit 3 is the input and qubits 1 and 5 are measured in the X basis. b) Effect of local complementation (LC) on qubit 5; c) LC on qubit 1; d) Pauli Z deletion of qubit 1; 3) LC on qubit 5; f) Pauli Z deletion of qubit 5. The final pattern represents the one-way two-qubit implementation of the $J(-\theta)$ gate, see Fig. 37.

the manipulations that follow we will use the following identities:

$$SXS^\dagger = Y \tag{7.13}$$

$$CYC^\dagger = Z \tag{7.14}$$

$$CXC^\dagger = X \tag{7.15}$$

Let us now consider two different graph states $|\psi\rangle$ and $|\phi\rangle$, specified below by listing their stabilizers, respectively:

$$\{f_1 = X_1Z_2Z_3, f_2 = Z_1X_2Z_3, f_3 = Z_1Z_2X_3\}, \tag{7.16}$$

$$\{f'_1 = X_1Z_2Z_3, f'_2 = Z_1X_2, f'_3 = Z_1X_3\}. \tag{7.17}$$

It is easy to check that the unitary $C_1S_2^\dagger S_3^\dagger$ applied on state $|\phi\rangle$ changes the stabilizers as follows:

$$\{g_1 = X_1Z_2Z_3, g_2 = Y_1Y_2, g_3 = Y_1Y_3\}. \tag{7.18}$$

Note that the new stabilizers satisfy the relations $f_1 = g_1$, $f_2 = g_1g_2$ and $f_3 = g_1g_3$, hence $C_1S_2^\dagger S_3^\dagger|\phi\rangle = |\psi\rangle$. The application of local one-qubit unitaries such as C and S does not change a state's entanglement structure, but reveals that different one-way patterns can correspond to the same entanglement resource. In our example, we know states $|\psi\rangle$ and $|\phi\rangle$ satisfy the eigenvalue equations $f_i|\psi\rangle = |\psi\rangle$ and $g_i|\phi\rangle = |\phi\rangle$. It is straighforward to verify that these states can be represented by the two sequences of commands $|\psi\rangle = CZ_{12}CZ_{13}CZ_{23}N_1N_2N_3$ and $|\phi\rangle = CZ_{12}CZ_{23}N_1N_2N_3$, where N_i is the preparation of qubit i in state $|+\rangle$. Now we are able to illustrate graphically the local complementation rule we described above by constructing the graphs associated with the states $|\psi\rangle$ and $|\phi\rangle$ and relating them by the equality $C_1S_2^\dagger S_3^\dagger|\phi\rangle = |\psi\rangle$, as shown in Fig. 43.

The local complementation operation on non-input qubit i corresponds to applying the unitary $C_i \prod_{j\in N(i)} S_j^\dagger$, with $N(i)$ being the set of vertices (qubits) which are neighbors of i in the entanglement graph. In addition to local complementation, we can also delete a vertex from a graph by measuring it in the Z basis [97]. Z-deletion and local complementation together change a state without altering the computation being performed [65, 97, 60, 107]. In general, we will need to apply these operations in graphs with an arbitrary number of vertices and edges, choosing where the local complementation is needed and applying the rule accordingly.

In Fig. 42 we illustrate a sequence of local complementations and Z-deletions that transforms the initial sequence (7.11) (corresponding to the circuit of Fig. 39) into a

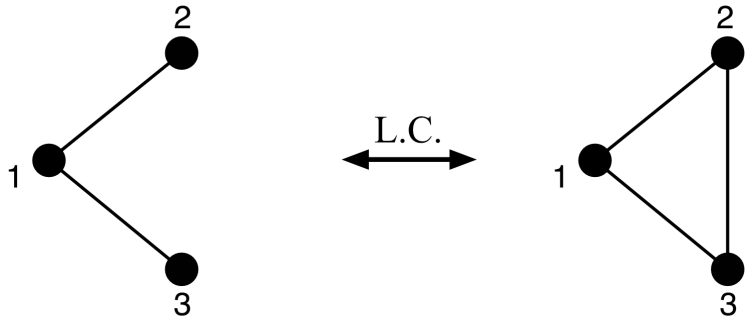


Figure 43: An analysis of the stabilizers in eqs. (7.16) and (7.17) indicates that the graph on the right represents state $|\psi\rangle$ and the one on the left state $|\phi\rangle$ (see main text). The local complementation unitary $C_1 S_2^\dagger S_3^\dagger$ changes the graph on the left into the graph on the right. Local complementations change the state without changing its entanglement structure or the one-way computations implementable by it.

simpler sequence implementing the same unitary. Starting from Fig. 42-a, we apply a sequence of local complementations operations in order to remove the postselected X-measurements originated from the BSS protocol. The required operations are illustrated in Fig. 42, so that in the end the postselected X-measurements become postselected Z-measurements, which can be removed. This procedure results in the following, equivalent command sequence:

$$P_3^{|\theta\rangle} CZ_{34} N_2 N_4 |\psi_{in}\rangle_3 = P_3^{|\theta\rangle} N_2 |G'\rangle = M_3^\theta Z_3^{s_3} N_2 |G'\rangle \tag{7.19}$$

where $|G'\rangle = CZ_{34}N_4|\psi_{in}\rangle_3$. Qubit 2 does not participate in the computation, as it remains disentangled from the others. Therefore, the final projection-based pattern is (written using $P_i^{|\theta\rangle} = M_i^\theta Z_i^{s_i}$):

$$M_3^\theta Z_3^{s_3} |G'\rangle. \tag{7.20}$$

We see that elimination of the postselected X measurement results in exactly the same command sequence of Eq. (7.2). As we have already seen, graph G' has regular flow and the associated measurement pattern is $X_4^{s_3} M_3^\theta |G'\rangle$, which implements deterministically the unitary map $J(-\theta)$. To conclude the protocol given in Def. 23 we can translate the computation back to the circuit model using the extended translation, which gives the circuit in Fig. 12-a. The compact circuit implementing the same unitary can be obtained by applying the compactification algorithm for regflow patterns (Alg. 1), which gives as result the circuit in Fig. 12-b. With this, we have obtained a time-respecting, perfectly physically implementable circuit from a CTC-assisted one.

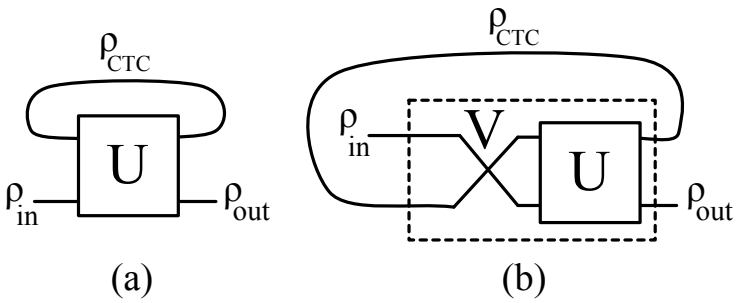


Figure 44: a) Deutsch's model for a CTC. b) This is the relationship between Deutsch's unitary U and unitary V in the BSS CTC circuit of Fig. 34-b.

7.5 Comparing the BSS model with Deutsch's model

In this section I review the CTC model proposed by David Deutsch [39] and compare the results obtained by applying BSS model to the circuit in Fig. 39 with the predictions provided by Deutsch's model when analyzing the same circuit. By doing so, I show that Deutsch's predictions are inconsistent with BSS's.

7.5.1 The Deutsch model

In 1991 Deutsch [39] proposed a different, highly influential model for CTCs in quantum theory. Deutsch's model avoids paradoxes by demanding self-consistent solutions for the time-travelling systems. Let us start by analyzing a simple CTC-assisted circuit, composed by just one time-respecting qubit and one time-travelling qubit, illustrated in Fig. 44-a, where U is a general two-qubit unitary. The correspondence with BSS model is shown in Fig. 44-b, so that Deutsch's $U = V \cdot SWAP$, with V being the unitary in BSS's formulation of CTCs (Fig. 34).

No paradox arises if we demand that the time-travelling qubit state ρ_{CTC} be a fixed point of the dynamics:

$$\rho_{CTC} = Tr_{TR}\left[U(\rho_{CTC} \otimes \rho_{in})U^{\dagger}\right], \tag{7.21}$$

where the partial trace is over the time-respecting qubit. This self-consistency requirement defines multiple solutions for ρ_{CTC}, each of which corresponds to a (generally non-linear) map on ρ_{in}, which can be worked out from the solution ρ_{CTC}:

$$\rho_{out} = Tr_{CTC}\left[U(\rho_{CTC} \otimes \rho_{in})U^{\dagger}\right]. \tag{7.22}$$

where the partial trace is over the time-traveling qubit.

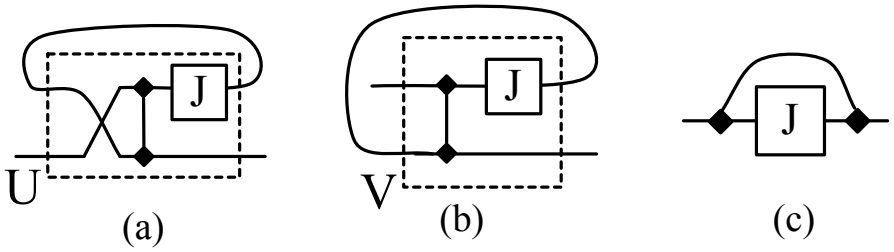


Figure 45: Three representations for the same CTC circuit. a) Deutsch formulation. b) BSS formulation. c) Short-hand form of either.

There exists at least one fixed point to Eq. (7.21) and therefore there is always a ρ_{CTC} satisfying the dynamics. This is so because every completely-positive map of the form $\mathcal{L}[\rho] = Tr_A[U(\rho \otimes \rho_A)U^\dagger]$ always has at least one solution. For scenarios where the self-consistency condition in Eq. (7.21) has more than one solution, Deutsch has postulated a "maximum entropy rule" stating that the solution with maximum entropy should be the one chosen.

Moreover, note that although the self-consistency equations preserve the state of the time-traveling qubit, the same doesn't happen to its correlations: the qubit that emerges into the past (after traveling through a CTC) does not preserve the (possibly) existing correlations with qubits elsewhere in the Universe. It is conceptually different from what happens in the BSS model, where both state and correlations of the time-traveling qubit are preserved. As a consequence, computers interacting with Deutschian CTCs are considerably more powerful than computers interacting with BSS CTCs. A quantum computer with access to Deutschian CTCs would be able to solve PSPACE-complete problems [7], a complexity class that strictly contains PP-complete problems - the type of problems that a quantum computer interacting with BSS CTCs is able to solve.

7.5.2 Comparison with the BSS model

Let us now study the same CTC we analyzed using BSS model, but now using Deutsch's by setting $U = [J(-\theta) \otimes 1] \cdot CZ \cdot SWAP$ in Fig. 44. In the BSS model this corresponded to the circuit in Fig. 36. Three graphical representations for the same CTC are shown in Fig. 45. We can represent $\rho_{in}(\vec{n}) = 1/2(1 + \vec{n} \cdot \vec{\sigma})$ and $\rho_{CTC}(\vec{m}) = 1/2(1 + \vec{m} \cdot \vec{\sigma})$, using the Pauli matrices $\vec{\sigma} = (X, Y, Z)$. A simple calculation using Eqs. (7.21) gives us the consistency conditions:

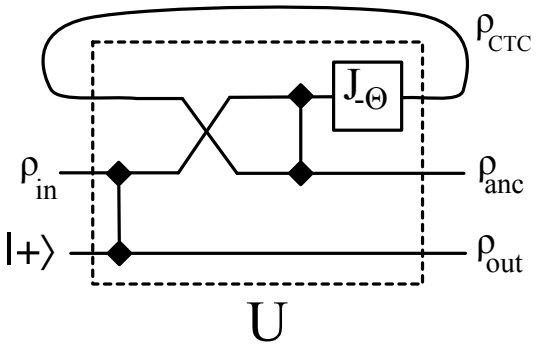


Figure 46: Deutsch's formulation of the extended CTC circuit of Fig. 38-b.

$$m_x = n_z, \tag{7.23}$$

$$m_y = m_z(n_x \sin\theta - n_y \cos\theta), \tag{7.24}$$

$$m_z = m_z(n_x \cos\theta + n_y \sin\theta). \tag{7.25}$$

The output state is $\rho_{out}(\vec{r}) = 1/2(1 + \vec{r}\cdot\vec{\sigma})$, with $\vec{r} = (m_x n_z, m_y n_z, m_z)$ being a function of self-consistently assigned m_x, m_y, m_z.

There are two classes of self-consistent solutions to Eqs. (7.23)-(7.25). The first is obtained by setting $m_z = 0$, which yields a unique self-consistent ρ_{CTC} for each input state:

$$\rho_{CTC} \;:\; \vec{m} = (n_z, 0, 0), \tag{7.26}$$

$$\rho_{out} \;:\; \vec{r} = (n_z^2, 0, 0). \tag{7.27}$$

These solutions are valid for all input states $\rho_{in}(\vec{n})$. The second class of solutions is obtained by assuming that $m_z \neq 0$ in Eqs. (7.23)-(7.25). Self-consistency dictates that such solutions exist only for the particular $\rho_{in} = |+_\theta\rangle\langle+_\theta|$, with $\rho_{CTC} = \rho_{out}$ described by $\vec{m} = (0, 0, m_z)$.

Our analysis of the BSS model for this CTC considered not only the circuit where the CTC appears on its own (Fig. 45), but also an enlarged circuit where the CTC acts on only part of a larger entangled state (Fig. 38). For a fair comparison between the two models, this can be done also in the Deutsch model, see Fig. 46. In this second approach Deutsch's unitary U is a three-qubit unitary encompassing all gates in the circuit of Fig. 38-b. A straightforward calculation using the self-consistency conditions in Eqs. (7.21) and (7.22) yields the solution in which ρ_{CTC} and ρ_{out} are parametrized by $\vec{m} = (n_z, 0, 0)$, with the ancilla qubit in output state $\rho_{anc}(\vec{a}) : \vec{a} = (n_z^2, 0, 0)$.

Our results for both the smaller circuit of Fig. 45-a and the larger circuit in Fig. 46 show that Deutsch's model fails to implement the same input-output map as BSS and the one-way model. The map implemented is the same only for the particular input state $\rho_{in} = |0\rangle\langle 0|$, with $\rho_{CTC} = \rho_{out} = |+\rangle\langle +|$. For this input state Deutsch's CTC qubit is not entangled with the time-respecting qubits.

This suggests that the root of the problem with Deutsch's model is the incompleteness of the description of the CTC qubit. Deutsch's model prescribes that the CTC qubit be sent back in time as a mixed density matrix, which results in information loss about its prior interactions. This is naturally taken care of in BSS model, as teleportation preserves the CTC qubit's entanglement with other systems, which was created via the unitary interaction V.

In the recent papers [92, 93] Ralph and Myers have proposed an extension of Deutsch's model, in which one can heuristically describe the situation from the point of view of the time-travelling qubit, which interacts with an infinite number of copies of itself. By adding decoherence to this system, a unique solution is selected out of Deutsch's possibly many self-consistent solutions. This extension of Deutsch's model is not, however, sufficient to make it compatible with the predictions of the BSS model and one-way quantum computation. A simple way to see that is to note that according to BSS the CTC output is always in a pure state, whereas the multiple solutions proposed by Deutsch are typically mixed.

7.5.3 Discussion

I have worked out the predictions of Deutsch's and BSS models for the same CTC examples and found a general disagreement between Deutsch's predictions and what is expected from the BSS/one-way model. This incompatibility stems from Deutsch's incomplete description of the state being sent back in time, whose complete history of previous interactions is preserved by the teleportation step used in BSS model.

One of the main conceptual differences between Deutsch's and BSS models are the self-consistency conditions. Deutsch's model imposes that the density matrix description of the qubit entering the CTC should be the same as the one that exits it (in the past). Therefore, a measurement performed in the time-traveling qubit in the future (entering the CTC) yields the same measurement statistics as if the measurement is performed in the same qubit in the past (that is, entering the CTC). On the other hand, the BSS model demands not only that such measurements yield the same statistics for the time-traveling

qubit but also that it should preserve its correlation with any other time-respecting qubits. In other words, if a qubit interacts with another qubit and becomes correlated with it before entering a CTC, the time-traveling qubit will exhibit the same correlations with the time-respecting qubit when it exits the CTC in the past.

The different self-consistency requirements in those models also yield different computation capabilities. The BSS model, which simulates the action of CTCs by performing quantum measurements and then post-selecting the results, can solve problems in the complexity class called `post-BQP`, or postselected quantum polynomial time. By the `post-BQP` = `PP` theorem [5], we say that BSS CTCs can solve `PP`-complete problems, which are a considerably powerful complexity class that strictly contains, for instance, the `NP` class. On the other hand, Deutsch's self consistency conditions requires Nature to solve a fixed point problem, which is known to be a `PSPACE`-complete problem. Therefore, Deutschian CTCs would in principle allow us to solve any problem in `PSPACE`, which properly contains the complexity class `PP` (class of problems that a BSS CTC would be able to solve).

In [75], Lloyd *et al.* worked out a link between CTCs as modeled by the BSS model and path-integral formulations of quantum mechanics. Indeed, path-integral formulations are specially suited for describing quantum field theories in curved spaces, a fact that resonates with the idea that the BSS model is more adequate for describing the power and action of CTCs. In particular, Lloyd *et al.* show in [75] that the dynamics provided by the BSS model coincides with the path-integral description of fermions using Grassmann fields given by Politzer [91]. In contrast, Deutsch's model predictions are incompatible with Politzer's path-integral approach [91].

7.6 One-way model: a toy model for space-time?

I would like to conclude this chapter with an interesting possible application of the 1WQC model, namely its use as a toy model for spacetime. One of the main problems in modern physics is the attempt to unify quantum mechanics, which describes three of the four known fundamental interactions (electromagnetism, weak nuclear force and strong nuclear force), with general relativity, which describes the forth fundamental interaction, gravity. Some of the approaches to unify those theories suggest that spacetime would not be independently constructed but rather a consequence of the laws of quantum mechanics. And it is in the line of those approaches that the one-way model for quantum computing

might have some role to play.

In [98, 99], R. Raussendorf *et al.* argue that the one-way quantum computation model has some of the properties which are expected in toy models able to generate spacetime structures from none. Firstly, time in 1WQC appears as binary relations (which induces the partial order) between quantum measurements (spacetime events), defined according to the geometry of the associated graph. Therefore, time is not an external parameter in the one-way model. Secondly, the correcting strategies defined by the flow theorems (Sec. 3.3) have as their main purpose guaranteeing that the logical processing of the computation is not affected by the inherent randomness of quantum measurements. In other words, a quantum mechanical principle, namely the random character of quantum measurements, is what induces the time-ordering in the one-way model.

Consider the following question: given a resource state $|G\rangle$ (a graph state), how much of temporal order can be defined directly from its geometry? If $|G\rangle$ is obtained from a quantum circuit (using, for instance, the methods described in Chapter 4), the temporal order can be easily inferred by comparison with the gate ordering in the associated quantum circuit. Let us suppose, however, that $|G\rangle$ is given with no reference to any kind of computation. As we have seen many times in this thesis, if the input and output sets are specified then the temporal order can be fully determined (by a flow theorem; See Sec. 3.3), even though we have not yet decided which computation is going to be performed using the resource state. However, not every (I, O) pair induces a possible partial order. In [80], the authors provide a way to determine which (I, O) pairs lead to a well-defined temporal structure. In other words, time in a 1WQC (partial order between measurement events) is almost entirely imposed by the geometry of the underlying graph; the missing element being just a specification of the set of first qubits to be measured (inputs) and the set of the last qubits to be measured (outputs).

One of the main results in [99] is a 1WQC analogue of Malament's theorem [77], which states that in any spacetime manifold the light cones determine the metric up to a conformal factor. In the 1WQC model, the computation is usualy specified by three elements: (i) resource state, (ii) measurement angles and (iii) classical processing relations, which give the partial order between measurement events. This specification can be further reduced since Theorem 4 in [99] states that (iii) completely determines (i). That is, the classical processing relations uniquely defines the resource state. Hence, the classical processing relations uniquely determines a one-way computation, up to the measurement angles. This is analogous to the Malament's theorem if the following identifications can

be inferred: (a) The light cones in general relativity correspond to classical processing relations (that is, the measurement outcomes that a qubit i depends on is the backward cone of i and the set of measurements that depend on qubit i's measurement outcome is the forward cone of i) and (b) the conformal factors in the spacetime metric at every spacetime point are the angles of the measurements performed at every measurement location (graph vertices).

In summary, the one-way model is able to mimic some of the properties of general relativity. A 1WQC analog of Malament's theorem was given in [99], where the authors also discuss backward and forward light cones (mainly based on flow-like set of conditions) and horizon of events. Those identifications between 1WQC and general relativity were based in the principle that the randomness of measurement outcomes should not affect the computation being performed, that is, the logical processing. It remains as an open question, however, to determine what could be the analogue of this principle in more general contexts, other than quantum computing. My work on closed timelike curves in the one-way model [41] (which predated [99]), explored in this chapter, goes in the same direction as [99], suggesting that 1WQC is indeed a good candidate as toy model for (at least some) general relativity phenomena.

8 Conclusion and further research directions

In this thesis I presented several results regarding the translation between the one-way quantum computation model and the circuit, gate-array model. By proposing a different approach for the translation of measurement patterns to the circuit model, I developed a complete optimization procedure that works for arbitrary quantum circuits. Moreover, using 1WQC techniques I have contributed to the discussion about the power of CTC-assisted quantum circuits, suggesting that the little known (at the time of [41]) CTC model by Bennett, Schumacher and Svetlichny gives more consistent predictions than the one proposed by Deutsch in his highly influential paper [39].

One of the goals I tried to achieve in this thesis is to give a more complete introduction to the so-called flow theorems. Those theorems characterize which graph states are needed to implement a given deterministic computation and give a prescription for doing it. It is clear that those theorems - and, more generally, the whole concept of deterministic computation in measurement-based models - constitute an important piece of knowledge for everyone interested in implementing 1WQC algorithms in physical systems. However, the flow theorems are little known in the physics community, mainly because most of the papers on the subject are published in computer science journals (and therefore written for a computer science audience). I hope I have contributed to make this subject more accessible and interesting to the physics community in Chapters 2 to 4, where I give an introduction to the 1WQC model with special attention to the determinism theorems.

In a series of results the key concepts of regflow, gflow, maximally delayed gflow, focused gflow and information preserving flow were introduced [33, 35, 24, 81, 80]. They address the general question of determinism in MBQC while exploring different forms of optimizations allowed by the model. I continued this line of research by introducing a new type of flow - the signal-shifted flow [42] - and exploring several of its structural properties. One of the most interesting properties of SSF is its parallel structure, which

equals in terms of depth the maximally delayed generalized flow [81]), which is the optimal flow. It is worth noting that the link between signal-shifted flow and maximally delayed gflow leads to a new efficient procedure for finding the optimal gflow of graphs with regflow, since the algorithm for finding the SSF of a graph with regflow is more efficient than the one for finding the maximally delayed gflow.

In Chapter 5, I propose a new way of translating 1WQC algorithms into quantum circuits. In this new framework I first obtain the extended translation of the measurement pattern one wants to translate, and then apply the so-called compactification procedure. A compactification procedure explores the correcting structure of the translated measurement pattern in order to rewrite the extended circuit, allowing the removal of several wires. Therefore, since the correcting structure of a measurement pattern is associated to the type of flow the associated graph satisfies, a compactification procedure must be derived for each type of flow. In Sec. 5.2, I analyzed the structural properties of the regflow function and designed an algorithm able to implement a compactification procedure for regflow extended circuits (Algorithm 1). Then, in Sec. 5.3, I did the same for the recently introduced signal-shifted flow, resulting in a compactification procedure for SSF extended circuits (Algorithm 4).

I have also explored a few examples of more general measurement patterns, namely the patterns associated with gflows that are different from regflow and SSF. Two complete examples are worked out in Sec. 5.4, where I also discuss the difficulty in designing a compactification procedure for arbitrary gflows; the existence of such a general compactification procedure remains as an interesting open question. Moreover, my scheme fails to compactify the parallel pattern obtained via Pauli simplification rules [23]. In other words one needs to keep the extra space in order to preserve the parallel depth obtained due to the Pauli measurements. This further indicates the crucial role that Clifford computation plays in obtaining the superior parallel power of 1WQC over quantum circuits.

The SSF compactification algorithm was designed in such a way that it preserves the reduced number of time slices $\mathcal{J}$ of SSF extended circuits. This allow us to use it for optimizing quantum circuits in the following way. First, we translate a circuit to the one-way model (as a regflow pattern) and then optimize it using the signal-shifting rules. The optimized pattern is then translated to the circuit model as a SSF extended circuit, from which one can obtain a compact circuit using our SSF compactification algorithm (Algorithm 4). This optimization procedure, called Signal Shifting Circuit Optimization (SSCO), is introduced in Chapter 6, where a complete example is worked out. Therein, I

use other translation and optimization techniques to compose an automated optimization procedure that explores the circuit's global structure to parallelize several J-gates and to group together several CX gates, which allow further optimizations to take place using specific method for Clifford gate parallelization [84].

Finally, in Chapter 7, I have shown how CTCs appear in a natural way in the one-way model of measurement-based quantum computation. I studied a simple example of such CTCs using the Bennett/Schumacher/Svetlichny (BSS) CTC model, whose predictions agreed with those required by the one-way model. Going beyond the simple example we studied, I have characterized a class of CTC circuits that admit deterministic BSS model simulations. The simulations can be found using stabilizer techniques associated with the one-way model, *i.e.*, using the flow theorems. I have also worked out the predictions of Deutsch's model for the same CTC example and found a general disagreement in comparison with what is expected from the BSS/one-way model. This incompatibility stems from Deutsch's incomplete description of the state being sent back in time, whose complete history of previous interactions is preserved by the teleportation step used in BSS model.

In the future, it would be interesting to characterize directly in the circuit model the class of CTC-assisted circuits for which an efficient description using only time-respecting qubits is possible. That could be found by re-interpreting in terms of circuit elements the (necessary and sufficient) conditions for determinism required by the generalized flow. Thus, instead of translating CTC-assisted circuits to the 1WQC model and then verifying if the associated graphs satisfy the gflow conditions, one could determine directly in the circuit model whether a CTC-assisted circuit could be deterministically simulated.

It would be interesting to apply the automated circuit optimization procedure developed in Chapter 6 to known quantum algorithms, comparing the gains with other known optimization methods unrelated to MBQC.

In more general terms, an interesting problem that could benefit from our translation approach is the study of restricted classes of computation. Recently, the study of classes of computations that lie between the classical and quantum - that is, non-universal classes of quantum computations that cannot be simulated efficiently in a classical computer - has become an important line of investigation; examples of those classes are the class of *instantaneous quantum computation* (IQP) [115] and those associated to the *Boson Sampling* problem [116, 117, 118, 119, 120]. It is an interesting exercise to see how the different restrictions translate between models, which may bring new insights on the

problem of quantum simulability.

An example where the translation to the 1WQC model has led to interesting results is the class of Clifford computations, which can be implemented in a single computational time slice in the one-way model. On the other hand, it is known that there are examples of single-round computations in the one-way model which do not translate as Clifford circuits. Hopefully, the translation methods developed in this thesis can aid in understanding better these new classes of interesting computations with finite depth.

Finally, as I have already mentioned, the development of a compactification procedure able to preserve (at least partially) the optimization provided by more general correcting strategies - in special those associated to arbitrary generalized flows - remains as an open question. It is clear that such a procedure would help in attacking the problems that I have raised here.

[1] C. H. Bennett and B. Schumacher. Unpublished. See http://web.archive.org/web/20030809140213/http://qpip-server.tcs.tifr.res.in/ qpip/HTML/Courses/Bennett/TIFR5.pdf (2002).

[2] D. M. Greenberger, M. A. Horne, A. Zeilinger. Going Beyond Bell's Theorem. *Bell's Theorem, Quantum Theory, and Conceptions of the Universe', M. Kafatos (Ed.), Kluwer, Dordrecht*, 69–72 (1989).

[3] D. Gottesman. Stabilizer Codes and Quantum Error Correction. *PhD Thesis* (1997).

[4] G. Chiribella, G. M. D'Ariano, P. Perinotti, B. Valiro.n Beyond causally ordered Quantum Computers. *arXiv:0912.0195* (2009).

[5] S. Aaronson. Quantum Computing, Postselection, and Probabilistic Polynomial-Time. *Proceedings of the Royal Society A*, vol. 461, 2063, 3473-3482 (2004).

[6] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70(5):052328 (2004).

[7] S. Aaronson and J. Watrous. Closed Timelike Curves Make Quantum and Classical Computing Equivalent. *Proc. R. Soc. A*, vol. 465, 2102, 631-647 (2009).

[8] I. Affleck, T. Kennedy, E. H.. Lieb, and H. Tasaki. Rigorous results on valence-bond ground states in antiferromagnets. *Phys. Rev. Lett.*, 59(7):799–802 (1987).

[9] D. Ahn, T. C. Ralph, and R. B. Mann. Any quantum state can be cloned in the presence of closed timelike curves . *arXiv:1008.0221*, Unpublished (2010).

[10] J. Anders and D. E. Browne. Computational power of correlations. *Phys. Rev. Lett.*, 102(050502), (2009).

[11] J. Anders, D. K. L. Oi, E. Kashefi, D. E. Browne, and E. Andersson. Ancilla-driven universal quantum computation. *Phys. Rev. A*, 82(2):020301 (2010).

[12] D. Bacon. Quantum computational complexity in the presence of closed timelike curves. *Phys. Rev. A*, 70(3):032309 (2004).

[13] J. T. Barreiro, M. Müller, P. Schindler, D. Nigg, T. Monz, M. Chwalla, M. Hennrich, C. F. Roos, P. Zoller, and R. Blatt. An open-system quantum simulator with trapped ions. *Nature*, 470(7335):486–491 (2011).

[14] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Phys. Rev. Lett.*, 70(13):1895–1899, (1993).

[15] C. H. Bennett, D. Leung, G. Smith, and J. A. Smolin. Can closed timelike curves or nonlinear quantum mechanics improve quantum state discrimination or help solve hard problems? *Phys. Rev. Lett.*, 103(17):170502, (2009).

[16] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders. Efficient Quantum Algorithms for Simulating Sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371 (2006).

[17] J. Biamonte and P. Love. Realizable Hamiltonians for universal adiabatic quantum computers. *Phys. Rev. A*, 78(1):012352 (2008).

[18] W. B. Bonnor. The rigidly rotating relativistic dust cylinder. *Journal of Physics A: Mathematical and General*, 13(6):2121–2132, (1980).

[19] G. Brennen and A. Miyake. Measurement-Based Quantum Computer in the Gapped Ground State of a Two-Body Hamiltonian. *Phys. Rev. Lett.*, 101(1):010502, (2008).

[20] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. van den Nest. Measurement-based quantum computation . *Nature Physics*, 5(1):20 (2009).

[21] H. J. Briegel and R. Raussendorf. Computational model underlying the one-way quantum computer. *Quantum information and Computation*, 6(433) (2002).

[22] A. Broadbent, J. F. Fitzsimons, and E. Kashefi. Universal Blind Quantum Computation . *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 517-526 (2009).

[23] A. Broadbent and E. Kashefi. Parallelizing Quantum Circuits. *Theoretical Computer Science*, 410(26):2489–2510, (2009).

[24] D. E. Browne, E. Kashefi, M. Mhalla, and S. Perdrix. Generalized Flow and Determinism in Measurement-based Quantum Computation. *New J. Phys.*, 9:250, (2007).

[25] D. E. Browne, E. Kashefi, and S. Perdrix. Computational depth complexity of measurement-based quantum computation. In Proceeding of the Fifth Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2010). pages 35-46 (2010).

[26] Č. Brukner, Jian-Wei Pan, C. Simon, G. Weihs, and A. Zeilinger. Probabilistic instantaneous quantum computation. *Phys. Rev. A*, 67(3):034304, (2003).

[27] T. A. Brun. Computers with Closed Timelike Curves Can Solve Hard Problems Efficiently. *Foundations of Physics Letters*, Vol. 16, Issue 3, pp 245-253 (2003).

[28] T. A. Brun, J.Harrington, and M. Wilde. Localized closed timelike curves can perfectly distinguish quantum states. *Phys. Rev. Lett.*, 102(21):210402, (2009).

[29] T. A. Brun and M. Wilde. Perfect state distinguishability and computational speedups with postselected closed timelike curves . *Foundations of Physics*, vol. 42, no. 3, pp 341-361 (2012).

[30] T. A. Brun, M. M. Wilde, and Andreas Winter. Quantum state cloning using Deutschian closed timelike curves. arXiv:1306.1795, unpublished (2013).

[31] S. Carroll, E. Farhi, and A. Guth. An obstacle to building a time machine. *Phys. Rev. Lett.*, 68(3):263–266, (1992).

[32] G. Chiribella, G. M. D'Ariano, P. Perinotti, and B. Valiron. Beyond causally ordered Quantum Computers. *arXiv:0912.0195*, (2009).

[33] V. Danos and E. Kashefi. Determinism in the one-way model. *Phys. Rev. A*, 74:052310, (2006).

[34] V. Danos, E. Kashefi, and P. Panangaden. Robust and parsimonious realisations of unitaries in the one-way model. *Phys. Rev. A*, 72:064301, (2005).

[35] V. Danos, E. Kashefi, and P. Panangaden. The Measurement Calculus. *Journal of the ACM*, 54(2), (2007).

[36] N. de Beaudrap. Finding flows in the one-way measurement model. *Phys. Rev. A*, 77, 022328, (2008).

[37] S. Deser, R. Jackiw, and G. t'Hooft. Physical cosmic strings do not generate closed timelike curves. *Phys. Rev. Lett.*, 68(3):267–269, (1992).

[38] D. Deutsch. Quantum computation networks. *Procedings of the Royal Society*, A 425(73-90), (1989).

[39] D. Deutsch. Quantum mechanics near closed timelike lines. *Phys. Rev. D*, 44(10):3197–3217, (1991).

[40] Raphael Dias da Silva and Ernesto F Galvão. Compact quantum circuits from one-way quantum computation. *Physical Review A*, 88, 012319, (2013).

[41] Raphael Dias da Silva, Ernesto F. Galvão, and E. Kashefi. Closed timelike curves in measurement-based quantum computation. *Physical Review A*, 83(1):012316, (2011).

[42] Raphael Dias da Silva, E. Pius, and E. Kashefi. Global Quantum Circuit Optimization. *arXiv:1301.0351*, unpublished (2013).

[43] R. Duncan and S. Perdrix. Rewriting Measurement-Based Quantum Computations with Generalised Flow. *37th International Colloquium on Automata, languages and programming, (ICALP)*, Bordeaux, France, July 6-10, 2010, Proceedings, Part II, pages 285–296, (2010).

[44] A. Einstein, B Podolsky, and N. Rosen. Can Quantum-Mechanical Description of Physical Reality Be Considered Complete? *Physical Review*, 47(10):777–780 (1935).

[45] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda. A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem. *Science*, 292(5516):472–475 (2001).

[46] R. Feynman. Simulating Physics with Computers. *International Journal of Theoretical Physics*, vol. 21, issue 6-7,pp 467-488 (1982).

[47] A. Friedenauer, H. Schmitz, J. T. Glueckert, D. Porras, and T. Schaetz. Simulating a quantum magnet with trapped ions. *Nature Physics*, 4(10):757–761 (2008).

[48] J. Friedman, M. Morris, I. Novikov, F. Echeverria, G. Klinkhammer, K. Thorne, and U. Yurtsever. Cauchy problem in spacetimes with closed timelike curves. *Phys. Rev. D*, 42(6):1915–1930, (1990).

[49] J. C. Garcia-Escartin and P. Chamorro-Posada. Equivalent Quantum Circuits. *arXiv:1110.2998v1*, unpublished (2011).

[50] K. Gödel. An Example of a New Type of Cosmological Solutions of Einstein's Field Equations of Gravitation. *Reviews of Modern Physics*, 21(3):447–450, (1949).

[51] J. Gott. Closed timelike curves produced by pairs of moving cosmic strings: Exact solutions. *Phys. Rev. Lett.*, 66(9):1126–1129, (1991).

[52] D. Gottesman and I. Chuang. Quantum Teleportation is a Universal Computational Primitive. *Nature* 402, 390-393, (1999).

[53] D. Gottesman and J. Preskill. Comment on "The black hole final state". *JHEP0403:026*, page 4, (2004).

[54] D. Gross and J. Eisert. Novel Schemes for Measurement-Based Quantum Computation. *Phys. Rev. Lett.*, 98(22):220503, (2007).

[55] D. Gross, J. Eisert, N. Schuch, and D. Perez-Garcia. Measurement-based quantum computation beyond the one-way model. *Phys. Rev. A*, 76(5):052315, (2007).

[56] D. Gross, S. T. Flammia, and J. Eisert. Most Quantum States Are Too Entangled To Be Useful As Computational Resources. *Phys. Rev. Lett.*, 102(19):190501, (2009).

[57] L. K. Grover. A fast quantum mechanical algorithm for database search. In *the twenty-eighth annual ACM symposium*, pages 212–219, New York, USA. ACM Press (1996).

[58] D. Genkina, G. Chiribella and L. Hardy. Optimal Probabilistic Simulation of Quantum Channels from the Future to the Past . Phys. Rev. A 85, 022330 (2012).

[59] S. Hawking. Chronology protection conjecture. *Phys. Rev. D*, 46(2):603–611, (1992).

[60] M. Hein, W. Dür, J. Eisert, R. Raussendorf, M. Van den Nest, and H. J. Briegel. In *Proceedings of the International School of Physics "Enrico Fermi" on "Quantum Computers, Algorithms and Chaos"* **162** (2006).

[61] G. T. Horowitz and J. Maldacena. The black hole final state. *JHEP02(2004)008.*, (2004).

[62] P. Hoyer and R. Spalek. Quantum Circuits with Unbounded Fan-out. *Theory of Computing*, 1(5):81-103 (2005).

[63] S F Huelga, M B Plenio, and J A Vaccaro. Remote control of restricted sets of operations: Teleportation of Angles. *Phys. Rev. A*, 65(042316), (2002).

[64] S F Huelga, J A Vaccaro, A Chefles, and M B Plenio. Quantum Remote Control: Teleportation of Unitary Operations. *Phys. Rev. A*, 63(042303):5, (2001).

[65] R Jozsa. An introduction to measurement based quantum computation. *arXiv:quant-ph/0508124*, Unpublished (2005).

[66] E. Kashefi. Lost in Translation. *Proceedings of the Third International Workshop on Development of Computational Models* (2007).

[67] E. Kashefi, D. K. L. Oi, D. E. Browne, J. Anders, and E. Andersson. Twisted graph states for ancilla-driven quantum computation. Proceedings of the 25th Conference on the Mathematical Foundations of Programming Semantics (MFPS 25), ENTCS, Vol. 249, pp. 307-331 (2009).

[68] J. Kempe, A. Kitaev, and O. Regev. The Complexity of the Local Hamiltonian Problem. *SIAM Journal on Computing*, 35(5):1070–1097 (2006).

[69] K. Kim, M. S. Chang, S. Korenblit, R. Islam, E. E. Edwards, J. K. Freericks, G. D. Lin, L. M. Duan, and C. Monroe. Quantum simulation of frustrated Ising spins with trapped ions. *Nature*, 465(7298):590–593 (2010).

[70] W. Klobus, A. Grudka, A. Wojcik. Comment on "Information flow of quantum states interacting with closed timelike curves" . *Phys. Rev. A* 84, 056301, (2011).

[71] K. Lanczos. ber eine stationre Kosmologie im Sinne der Einsteinschen Gravitationstheorie. *Zeitschrift fur Physik*, 21(1):73–110, (1924).

[72] B P Lanyon, C Hempel, D Nigg, M Muller, R Gerritsma, F Zahringer, P Schindler, J T Barreiro, M Rambach, G Kirchmair, M Hennrich, P Zoller, R Blatt, and C. F. Roos. Universal Digital Quantum Simulation with Trapped Ions. *Science*, 334(6052):57–61 (2011).

[73] D. Leung. Two-qubit Projective Measurements are Universal for Quantum Computation. *NSF-ITP-01-174*, (2001).

[74] S. Lloyd. Universal Quantum Simulators. *Science*, 273(5278):1073–1078 (1996).

[75] S. Lloyd, L. Maccone, R. Garcia-Patron, V. Giovannetti, and Y. Shikano. The quantum mechanics of time travel through post-selected teleportation . *Phys. Rev. D*, 84(2):025007 (2010).

[76] S. Lloyd, L. Maccone, R. Garcia-Patron, V. Giovannetti, Y. Shikano, S. Pirandola, L. Rozema, A. Darabi, Y. Soudagar, L. Shalm, and A. Steinberg. Closed Timelike Curves via Postselection: Theory and Experimental Test of Consistency. *Phys. Rev. Lett.*, 106(4):040403, (2011).

[77] D. B. Malament. The class of continuous timelike curves determines the topology of spacetime. *Journal of Mathematical Physics*, 18(7):1399–1404 (1977).

[78] D. Markham, J. Anders, M. Hajdušek, and V. Vedral. Measurement Based Quantum Computation on Fractal Lattices. *Electronic Proceedings in Theoretical Computer Science*, 26:109–115, (2010).

[79] D Maslov, G W Dueck, D M Miller, and C Negrevergne. Quantum Circuit Simplification and Level Compaction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 27, Issue 3, pages 436-444 (2008).

[80] M. Murao, S. Perdrix, M. Someya, P. S. Turner, M. Mhalla. Which graph states are useful for quantum information processing? *Sixth Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC)*, arXiv:1006.2616 (2011).

[81] M Mhalla and S. Perdrix. Finding optimal flows efficiently. *35th International Colloquium on Automata, Languages and Programming, ICALP*, Proceedings, Part I, pp 857-868 (2008).

[82] M. Mhalla and S. Perdrix. Graph States, Pivot Minor, and Universality of (X,Z)-measurements. *International Journal of Unconventional Computing*, Vol. 9, Issue 1/2, p153 (2012).

[83] A. Miyake. Quantum Computation on the Edge of a Symmetry-Protected Topological Order. *Phys. Rev. Lett.*, 105(4):040501, (2010).

[84] C. Moore and M. Nilsson. Parallel Quantum Computation and Quantum Codes. *SIAM J. Computing*, 31:799–815, (2001).

[85] C. E Mora, M Piani, A. Miyake, M. van den Nest, W Dür, and H. J. Briegel. Universal resources for approximate and stochastic measurement-based quantum computation. *Phys. Rev. A*, 81(4):042315, (2010).

[86] M. A. Nielsen and I. L. Chuang. *Quantum computation and quantum information.* Cambridge University Press, (2010).

[87] M. Nielsen. Universal quantum computation using only projective measurement, quantum memory, and preparation of the 0 state. *Phys. Lett. A. 308 (2-3): 96-100*, (2003).

[88] M. Nielsen. Cluster-state quantum computation. *Reports on Math. Phys.* Vol. 57, Issue 1, Pages 147-161 (2006).

[89] M. Nielsen, D. Leung, and A. M. Childs. Unified derivations of measurement-based schemes for quantum computation. *Phys. Rev. A*, 71(032318):14 (2004).

[90] D T Pegg. Quantum mechanics and the time travel paradox. *Times' arrows, quantum measurement and superluminal behavior* (Eds. D. Mugnai, A. Ranfagni and L. S. Schulman) p. 113, (2001)

[91] H Politzer. Path integrals, density matrices, and information flow with closed timelike curves. *Phys. Rev. D*, 49(8):3981–3989, (1994).

[92] T C Ralph. Unitary solution to a quantum gravity information paradox. *Phys. Rev. A*, 76:012336, (2007).

[93] T C Ralph and C R Myers. Information Flow of quantum states interacting with closed timelike curves . *Phys Rev A*, 82, 062330 (2010).

[94] R. Raussendorf and H. J. Briegel. Quantum computing via measurement only. arXiv:quant-ph/0010033 (2000).

[95] R. Raussendorf and H. J. Briegel. A One-Way Quantum Computer. *Phys. Rev. Lett.*, 86(22):5188–5191, (2001).

[96] R. Raussendorf, H. J. Briegel, and D. E. Browne. The one-way quantum computer - a non-network model of quantum computation. *Journal of Modern Optics 49 1299*, (2002).

[97] R. Raussendorf, H. J. Briegel, and D. E. Browne. Measurement-based quantum computation on cluster states. *Phys. Rev. A*, 68(022312), (2003).

[98] R. Raussendorf, P. Sarvepalli, T. C. Wei, and P. Haghnegahdar. Measurement-based quantum computation–a quantum-mechanical toy model for spacetime? arXiv:1108.5774 (2011).

[99] R. Raussendorf, P Sarvepalli, Tzu-Chieh Wei, and P Haghnegahdar. Symmetry constraints on temporal order in measurement-based quantum computation . *Proceedings 8th International Workshop on Quantum Physics and Logic Nijmegen, Netherlands, October 27-29 (EPTCS 95)*, pp. 219-250 (2011).

[100] R. Raussendorf and Tzu-Chieh Wei. Quantum computation by local measurement. *Annual Review of Condensed Matter Physics*, vol. 3, pages 239-261 (2012).

[101] J. Roland and N. Cerf. Quantum search by local adiabatic evolution. *Phys. Rev. A*, 65(4):042308 (2002).

[102] A. C. Cem Say and A. Yakaryılmaz. Computation with narrow CTCs. *Unconventional Computation, Lecture Notes in Computer Science*, Vol. 6714, pp 201-211 (2011).

[103] Michal Sedlák and Martin Plesch. Towards optimization of quantum circuits. *Central European Journal of Physics*, 6:128–134, (2008).

[104] K. H. Shah and D. K. L. Oi. Ancilla Driven Quantum Computation with arbitrary entangling strength. *arXiv:1303.2066*, (2013).

[105] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509 (1997).

[106] G. Svetlichny. Effective Quantum Time Travel. *Int. J. of Theor. Phys.* 50, 3903 (2011).

[107] M. van den Nest, J. Dehaene, and B. De Moor. Graphical description of the action of local Clifford transformations on graph states. *Phys. Rev. A*, 69(022316):8, (2004).

[108] M. van den Nest, A. Miyake, W. Dür, and H. J. Briegel. Universal Resources for Measurement-Based Quantum Computation. *Phys. Rev. Lett.*, 97(15):150504, (2006).

[109] W. J. van Stockum. The gravitational field of a distribution of particles rotating around an axis of symmetry. *Proc. Roy. Soc. Edinburgh*, 57, pp. 135-154 (1937).

[110] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887 (2001).

[111] J. Wallman and S. D. Bartlett. Revisiting consistency conditions for quantum states of systems on closed timelike curves: an epistemic perspective . *Found. Phys.* 42, 656-673 (2012).

[112] Tzu-Chieh Wei, I. Affleck, and R. Raussendorf. Affleck-Kennedy-Lieb-Tasaki State on a Honeycomb Lattice is a Universal Quantum Computational Resource. *Phys. Rev. Lett.*, 106(7):070501, (2011).

[113] Tzu-Chieh Wei, I. Affleck, and R. Raussendorf. Two-dimensional Affleck-Kennedy-Lieb-Tasaki state on the honeycomb lattice is a universal resource for quantum computation. *Phys. Rev. A*, 86(3):032328, (2012).

[114] N. Yoran and A. J. Short. Classical simulation of limited-width cluster-state quantum computation . *Phys. Rev. Lett.*, 96(17):170503, (2006).

[115] D. Shepherd, M. J. Bremner. Temporally unstructured quantum computation. *Proc. R. Soc. A 465*, 1413-1439 (2009).

[116] A. Crespi, R. Osellame, R. Ramponi, D. J. Brod, E. F. Galvao, N. Spagnolo, C. Vitelli, E. Maiorino, P. Mataloni, F. Sciarrino. Integrated multimode interferometers with arbitrary designs for photonic boson sampling. *Nature Photonics* 7, 545 (2013).

[117] J. B. Spring, et al. Boson sampling on a photonic chip. *Science 339*, 798-801 (2013).

[118] M. Tillmann, B. Dakic, R. Heilmann, S. Nolte, A. Szameit and P. Walther. Experimental boson sampling. *Nature Photonics* 7, 54044 (2013).

[119] Photonic Boson Sampling in a Tunable Circuit. M. A. Broome, A. Fedrizzi, S. Rahimi-Keshari, J. Dove, S. Aaronson, T. Ralph, A. G. White. *Science 339*, 6121 (2013).

[120] S. Aaronson and A. Arkhipov. The computational complexity of linear optics. *Proceedings of the 43rd annual ACM symposium on Theory of computing* (STOC), pages 333-342 (2011).