Extended Abstracts Submitted by the AQUA Team to AQIS 2010

August 4, 2010

1. Takahiko Satoh and Rodney Van Meter, “Path Selection in Heterogeneous Quantum Networks”
2. Luciano Aparicio and Rodney Van Meter, “Multiplexing in Quantum Repeater Networks”
3. Byung-Soo Choi and Rodney Van Meter, “A Quantum Adder on a Two-Dimensional Qubit Array Architecture”
4. Clare Horsman, Simon Devitt, and Rodney Van Meter, “Algorithm Optimisation for Topological Measurement-Based Quantum Computing”
5. Cody Jones, Austin G. Fowler, Jungsang Kim, Thaddeus D. Ladd, Rodney Van Meter, and Yoshihisa Yamamoto, “A Layered Architecture for Quantum Computing using Optically-Controlled Quantum Dots”
6. Shota Nagayama and Rodney Van Meter, “Defective Qubits in Surface Code Quantum Computation on a Fixed Lattice”
7. Rodney Van Meter, Thaddeus D. Ladd, Austin G. Fowler, and Yoshihisa Yamamoto, “Heterogeneous Interconnects in a Semiconductor Nanophotonic Surface Code Quantum Computer”

Path Selection in Heterogeneous Quantum Networks

Takahiko Satoh[1] * Rodney Van Meter[2] †

[1] *Department of Computer Science, Graduate School of Information Science and Technology, the University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, Japan.*
[2] *Faculty of Environment and Information Studies, Keio University, 5322 Endo, Fujisawa-city, Kanagawa, Japan.*

Abstract. To date, research on entangled quantum networks has primarily focused on an abstract model consisting of a linear chain of repeaters, with a power of two number of hops of identical length and quality. We are analyzing the behavior of more complex network topologies. In such configurations, path selection affects both the performance of individual connections and global network load. We propose a definition for link cost and a form of Dijkstra's algorithm for ranking candidate paths to maximize the throughput of end-to-end connections. Simulations confirm agreement between the calculated path cost and the expected throughput.

Keywords: Quantum repeater, quantum network, path selection algorithm

1 Introduction

Networks of quantum repeaters use purification and entanglement swapping to create high-fidelity end-to-end Bell pairs which are then used in distributed quantum applications [1, 2]. In entanglement swapping, two distributed Bell pairs are spliced together where they meet, creating one longer-distance pair from two shorter ones. For homogeneous paths of 2^n hops, quantum repeaters operate well using a simple doubling of distance for each entanglement swap. In real-world networks, candidate paths will have differing numbers of hops of varying quality, as in Fig. 1. On such heterogeneous paths, we must reconsider when and where entanglement swapping takes place, and devise an algorithm for ranking paths. We use the inverse of single-hop throughput as our link cost. Path cost, as in Dijkstra's algorithm [3], is the sum of the link costs in the path. We have extended our quantum network simulator to calculate the throughput of end-to-end connections for heterogeneous links.

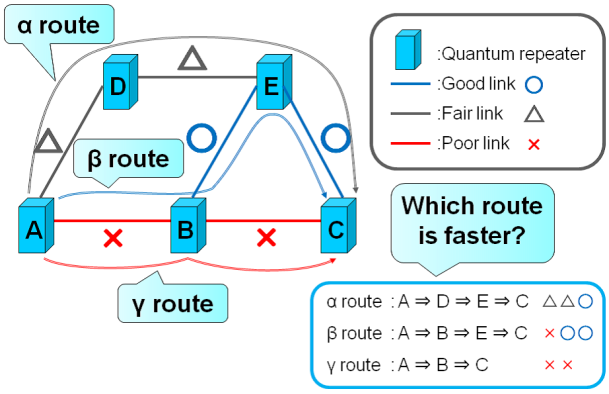


Figure 1: Real-world quantum networks will offer multiple heterogeneous paths between communication end points, necessitating a mechanism for path selection.

2 Background

2.1 Quantum Repeaters

Quantum repeaters use single-hop quantum links and a series of classical messages to forge end-to-end entangled Bell pairs. The responsibility for purification and swapping can be divided into layers (shown in Appendix A), communicating across different subsets of the repeaters in the path. The throughput is known to decline polynomially with distance, though the complexities of limited systems make it difficult to create an analytic expression for the throughput.

2.2 Networking

Path selection is important in all networks. In quantum networks, a small difference in path quality can cause a large difference in throughput. Dijkstra's algorithm is a powerful technique used extensively in the Internet for solving the path selection problem. In this algorithm, path cost is Σc_i, where c_i is the cost assigned to link i. The goal of Dijkstra is minimization of global work, rather than maximization of throughput for any specific communication session, but in practice there is a strong correlation between those two goals.

2.3 Problem Definition

Naturally, lower path quality will reduce both the fidelity of end-to-end entanglement and the throughput of a given path, but we do not know the exact relationship between path quality and throughput. We wish to define a cost for each path, allowing us to order a set of paths so that low-cost paths better meet our goals for the overall system. Here, we choose maximizing throughput as our goal, meaning that high-bandwidth paths (measured in high-fidelity Bell pairs per second) should have a low cost. To achieve this, we need both a way to establish

*satoh@is.s.u-tokyo.ac.jp
†rdv@sfc.wide.ad.jp

the cost for a single link, and a way to calculate path cost from a set of link costs.

3 Quantum Dijkstra

Our proposed algorithm is a direct adaptation of Dijkstra's algorithm to quantum networks. We have examined several possible candidates for link cost, including link base entanglement fidelity and loss (either linear or in dB). Our calculations suggest that defining link cost based on throughput when used as a single hop repeater works well. We normalize across the whole network, giving

$$Linkcost = \frac{maximum\ single\text{-}hop\ throughput}{single\text{-}hop\ purified\ throughput} \quad (1)$$

As in classical Dijkstra, our quantum Dijkstra (which we call qDijkstra) defines path cost as the sum of the link costs.

4 Results

In our simulations, we used four different link qualities with the qubus entanglement mechanism, which uses bright pulses and weak non-linear effects [4]. Qubus is highly sensitive to loss, so we simulate 20 km optical fiber links assuming 0.17 dB/km loss, plus an additional system loss factor.

4.1 Linkcost

To determine link cost, we simulated a single hop for a variety of loss values. We selected four values to be our link types, as shown in Table 1. For simplicity, as described above, we have normalized the link costs relative to the highest-throughput link. Further details are presented in Appendix B.

Table 1: Linkcost

path quality	dBloss (dB/20 km)	throughput (qubits/sec)	linkcost
Excellent (□)	3.4	176.7	1
Good (○)	3.5	137.9	1.28
Fair (△)	3.6	118.5	1.49
Poor (×)	3.7	95.6	1.84

4.2 Verification of algorithm

We tested sixteen types of 8-hop paths, with heterogeneous links in the first half of the path and homogeneous, excellent links in the second half, and compared simulated throughput and qDijkstra's cost. The results, plotted in Fig. 2, confirm a strong relationship between throughput and path score.

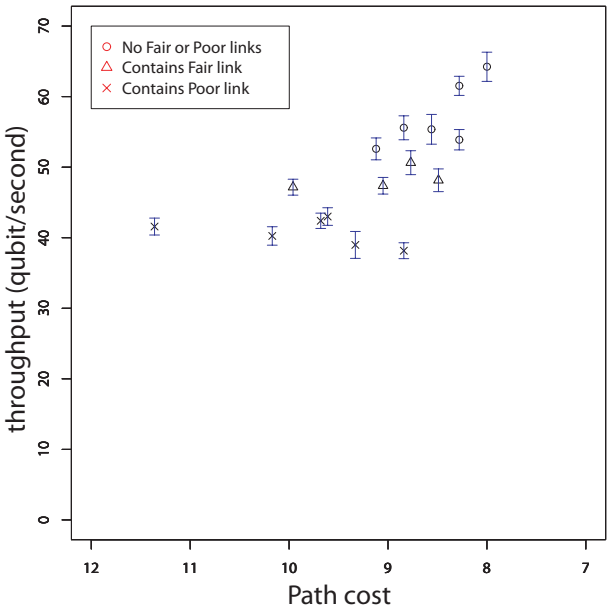


Figure 2: Throughput and the inverse of path cost for eight-hop paths. The symbols ◯, △ and × correspond to the weakest link in the path, good, fair, and poor links, respectively. Further details are presented in Appendix C.

5 Discussion

Our simulations show that classical networking techniques can be readily adapted to quantum repeater networks. As in classical networks, the throughput of a single path will approach the throughput of the lowest-quality link, as shown by the grouping of the symbols in Fig. 2. Dijkstra's algorithm, in both classical and quantum networks, gives reasonable but not perfect ordering of candidate paths. Using qDijkstra, we can accurately choose the best path without the prohibitive expense of enumerating and simulating all of the possible paths through large networks. These results give us reason for optimism about the difficulty of quantum network design and a clear direction for achieving that goal.

Acknowledgements

This work was supported by JSPS KAKENHI 21500020. The authors thank François Le Gall for useful discussions.

References

[1] H.-J. Briegel, W. Dür, J.I. Cirac, and P. Zoller. Quantum Repeaters: the Role of Imperfect Local Operations in Quantum Communication *Physical Review Letters*, volume 81, pages 5932–5935, 1998.

[2] C. H. Bennett, G. Brassard, C. Crépeau, R. Josza, and A. Peres, W. Wootters. Teleporting an unknown quantum state via dual classical and EPR channels

Physical Review Letters, volume 70, pages 1895–1899, 1993.

[3] Dijkstra, EW. A note on two problems in connexion with graphs. *Numerische Mathematik*, volume 1, number 1, pages 269-71. Springer, 1959.

[4] T. D. Ladd, P. van Loock, K. Nemoto, and W. J. Munro, Y. Yamamoto. Hybrid quantum repeater based on dispersive CQED interaction between matter qubits and bright coherent light *New Journal of Physics*, volume 8, pages 184, 2006.

[5] Rodney Van Meter, Thaddeus D. Ladd, W. J. Munro, and Kae Nemoto. System Design for a Long-Line Quantum Repeater. *IEEE/ACM Transactions on Networking*, volume 17, number 3, pages 1002–1013, June 2009. 10.1109/TNET.2008.927260.

A Quantum Network

A conceptual diagram of the protocol layers necessary for a quantum network is shown in Fig. 3 [5].

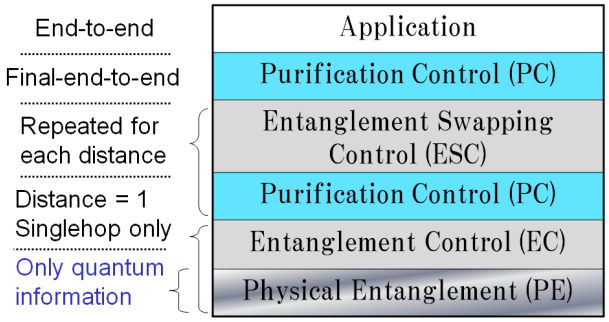


Figure 3: Protocol layers for quantum repeaters.

B Throughput of a Single Hop Path

Table 2 and Fig. 4 are the simulation environment and results used to establish our link cost for a single hop. The four link types chosen for more complex paths are marked in the figure with the corresponding symbols.

Table 2: Measurement environment of Linkcost

Quantum Repeater	
Number of qubits per repeater link connection	25 qubits
Number of application qubits teleported (length of simulation)	200 qubits
Optical Fiber	
Length	20 km
Information losses	3.4∼4.4 dB/20 km

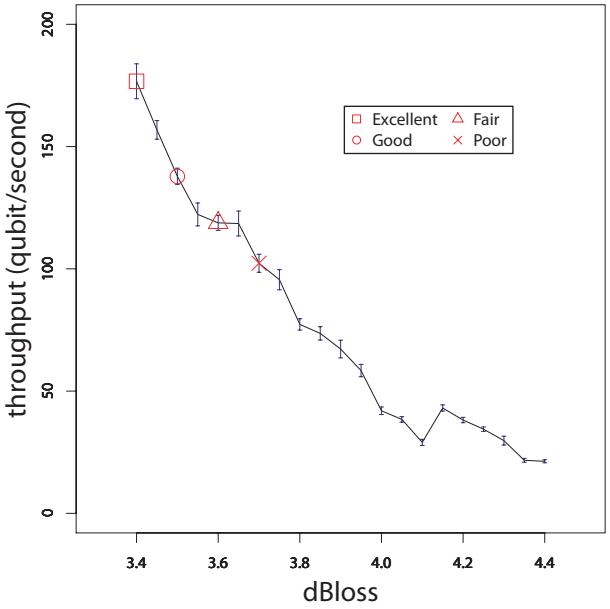


Figure 4: Single-hop simulation, used to define linkcost.

C Throughput of 8-hop Paths

The details of our simulated 8-hop paths are shown in Table 3. The string of symbols indicates the type and order of links in the path.

Table 3: Details of measurement

Path combination	Throughput	Path cost
□□□□□□□□	64.2278	8
□○□□□□□□	61.5261	8.28
○□□□□□□□	53.8885	8.28
□△□□□□□□	48.1412	8.49
□□○○□□□□	55.3664	8.56
□○□△□□□□	50.6342	8.77
□○○○□□□□	55.5797	8.84
□×□□□□□□	38.1576	8.84
○□○△□□□□	47.3658	9.05
○○○□□□□□	52.5923	9.12
□△×□□□□□	38.9788	9.33
○△□×□□□□	43.0007	9.61
×□×□□□□□	42.3994	9.68
△△△△□□□□	47.163	9.96
△××□□□□□	40.2496	10.17
××××□□□□	41.5825	11.36

Multiplexing in Quantum Repeater Networks

Luciano Aparicio[1] * Rodney Van Meter[2] †

[1] *Graduate School of Information Science and Technology, The University of Tokyo*
7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan.
[2] *Faculty of Environment and Information Studies, Keio University*
5322, Endo, Fujisawa, Kanagawa, 252-0882 Japan

Abstract. Existing research has investigated quantum repeater networks only in abstract, ideal settings. We are considering a more realistic approach and context, including complex technologies with multiple connections competing for resources. For these scenarios, we are comparing several different classical multiplexing schemes using simulation, measuring the aggregate throughput and fairness for each of them. Preliminary results suggest that round-robin use of a congested link gives the highest aggregate throughput.

Keywords: quantum networks, quantum repeaters, multiplexing

1 Introduction

Applications that use distributed quantum properties, such as QKD (Quantum Key Distribution), have the limitation that the fidelity of quantum states and the probability of success decrease with distance, making the usage of these systems for long distances almost impossible. Therefore, researchers have proposed the design of quantum repeater networks [1] which would maintain distributed quantum states across greater distances. Although some researchers are investigating approaches that are substantially different from entanglement swapping [2], here we focus on swapping.

In classical networks, stations are usually competing for shared resources, the service is offered on a best effort basis, and the path selection is a difficult task handled by many different routing algorithms. Quantum networks will not be an exception. Based on analogy with the classical model, to solve the shared resources issues, we propose quantum multiplexing and evaluate several schemes which we are simulating.

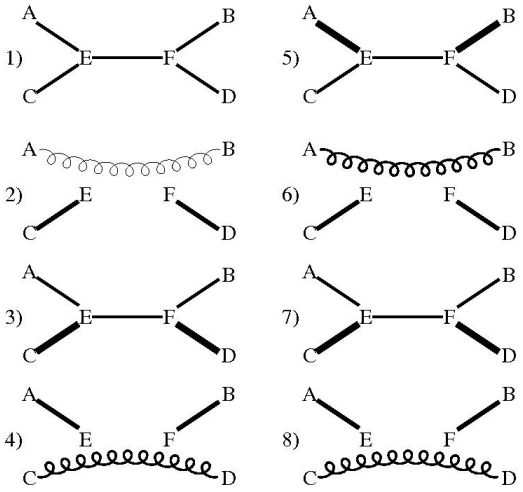


Figure 1: Multiplexed use of a congested link in a dumbbell network raises aggregate throughput compared to pure circuit switching.

2 Background

2.1 Quantum Networks

The process of designing quantum networks is similar to designing classical networks, as they require detailed protocol designs, including finite states machines to control physical resources (qubit entanglement, purification and entanglement swapping in order to maintain adequate fidelity and allow teleportation of qubits from one station to another). Proposed protocols [3] are composed of layers in analogy to the OSI model. We give a brief description of each of them and the functions which are simulated.

The physical layer represents the physical interaction that creates Bell pairs between two different stations. Our simulations model the qubus mechanism [4] in which laser pulses of many photons generate low-fidelity Bell pairs with high probability.

The second layer, EC (Entanglement Control), is responsible for managing the single-hop physical entanglement process, utilizing classical messages. Once the entanglement is confirmed, this layer will transfer control of the Bell pair to a higher protocol layer.

The third layer, PC (Purification Control), is responsible for the purification of the entangled pairs. For single hops, after the PC layer confirms a good fidelity for the Bell pairs, control is given to the application layer, which will start the teleportation of the data qubit that we wish to send.

For networks which have more than two stations, further steps are required. ESC (Entanglement Swapping Control) is the layer responsible for choosing two Bell pairs from two different stations and performing the Bell state measurement that splices the two short Bell pairs into a longer one. After this is concluded, once again the PC layer executes purification to improve the fidelity of this new Bell pair. ESC and PC are repeated until we have an end-to-end Bell pair of sufficient fidelity for our application.

*lucho@hongo.wide.ad.jp
†rdv@sfc.wide.ad.jp

2.2 Multiplexing

For a quantum network of more than two end nodes we may have to consider sharing a channel between different stations. Therefore, a way to control the resource allocation must be provided. In this work, we want to prove that classical multiplexing strategies can also be applied to quantum networks. Different multiplexing schemes, such as statistical multiplexing (packet switching), circuit switching, TDM (Time Division Multiplexing) and spatial multiplexing have been studied and compared.

Statistical multiplexing (packet switching) is used in IP networks, as a best effort service.

Circuit switching, used in standard telephony service, reserves a complete circuit for one connection and no other traffic is allowed to use those resources at the same time. When the connection is finished, the circuit is released, allowing other stations to transmit.

TDM will assign resources in a time slot round-robin fashion for each station that requests a connection. Every station will wait its turn to transmit, and if a station is not ready to use the channel on its turn, the resources go unused.

In spatial multiplexing, some resources (e.g., buffer space) are assigned to one connection.

3 Quantum Repeater Multiplexing

Each of the multiplexing schemes can be used fairly directly in repeater networks. The principal operational difference between classical and quantum is that entanglement swapping is a truly distributed stochastic computation, with state held at each station along the path. The natural model would therefore be circuit switching, but our analysis suggests that round-robin TDM on the congested link brings benefits.

Figure 1 shows a sequence of entanglement, purification and entanglement swapping steps, multiplexing use of the central link in the dumbbell network. Straight lines represent single-hop Bell pairs, and their thickness represents fidelity. In step 1, all the links have one low-fidelity Bell pair. In step 2, entanglement swapping is done, and A and B become entangled (spring line). Meanwhile, fidelity is improved for the links EC and FD using new base pairs and purification. In step 3, new Bell pairs are obtained for AE, EF and FB, while fidelity keeps increasing for CE and FD as in step 2. In step 4, entanglement swapping is done for C and D, but in this case with better end-to-end fidelity than in step 2 due to the higher fidelity on the uncongested links.

4 Simulations

For the case of the dumbbell network studied, one link is shared. This link has many resources (qubits to be used for creating Bell pairs), which allows us to implement different types of multiplexing schemes. In this topology, station A sends traffic to B, and at the same time station C sends traffic to D.

For the simulations, we are using Omnet++ 4.0, a C++-based network simulator. Each station is configured to have 8 qubits for each link connecting to other stations. Once single-hop Bell pairs are created, entanglement swapping is done in two places, in order to create an end-to-end Bell pair, which will be immediately used by the application.

We are simulating a restricted case of the qubus repeater. For 10km links with 0.17dB/km loss, simulations show that the basic qubus mechanism should produce base entangled pairs of F = 0.769 with probability 36%. Three rounds of purification will then improve fidelity for a single hop in the following manner: $(0.769{+}0.769) \Rightarrow 0.872, (0.872 + 0.872) \Rightarrow 0.951, (0.951 + 0.951) \Rightarrow 0.982$.

For each hop, three successful rounds of purification requires eight base-fidelity Bell pairs. Once entanglement swapping is done, fidelity drops to a value that is roughly the product of the fidelity of each hop. For two hops it would be around 0.964 and for three hops 0.947.

The parameters which are measured are the aggregate throughput (minimum time to complete the workload) and fairness (how much time one connection makes the others wait).

5 Discussion

From this research, we can see that classical multiplexing presents us with a possible approach to handle the problem of shared resources in quantum networks. Multiplexing use of congested links while uncongested links continue purification gives higher throughput than pure circuits switching. Our simulations are being extended to larger networks and more complex traffic patterns, which we believe will demonstrate more distinct differences between the types of multiplexing. Future work will also include quantum effects for the determination of the fidelity, instead of the classical aproximation of this work.

6 Acknowledgements

LA acknowledges funding from MEXT. This work was supported by JSPS KAKENHI 21500020.

References

[1] W. Dür, H.-J. Briegel, J. I. Cirac, P. Zoller. *Quantum repeaters based on entanglement purification.* Physical Review A, Volume 59, Number 1. 1999.

[2] W. J. Munro, K. A. Harrison, A. Stephens, S. Devitt, and K. Nemoto. *From quantum fusiliers to high-performance networks.* (2009) Arxiv preprint arXiv:0910.4038.

[3] Rodney Van Meter, Thaddeus D. Ladd, W. J. Munro, Kae Nemoto. *System Design for a Long-Line Quantum Repeater.* IEEE/ACM Transactions on Networking, Vol. 17, NO. 3, June 2009.

[4] P. van Loock, T. D. Ladd, K. Sanaka, F. Yamaguchi, K. Nemoto, W. J. Munro, and Y. Yamamoto. *Hybrid quantum repeater using bright coherent light.* Phys. Rev. Lett., vol. 96, p. 240501, 2006.

A Quantum Adder on a Two-Dimensional Qubit Array Architecture

Byung-Soo Choi[1] * † Rodney Van Meter[2] ‡

[1] *Department of Electronics Engineering, Ewha Womans University, Seoul, Republic of Korea (South Korea)*
[2] *Faculty of Environment and Information Studies, Keio University, Fujisawa, Japan*

Abstract. We propose a quantum adder on a two-dimensional qubit array architecture with nearest-neighbor interaction. The depth of our new adder is $O(\sqrt{n})$ where n is the size of input, which is asymptotically optimal. The size is $O(n)$. When the input size is larger than 58, our new adder can work faster than the simple embedding of ripple-carry adders designed for one-dimensional systems.

Keywords: Quantum Architecture, Quantum Adder, Depth Lower Bound, Interaction Distance

1 Motivation

Theoretically, quantum information processing systems will show higher performance than classical ones in some areas [1]. Although this property drives us to design and implement a scalable quantum computer, the current state of the art is far from achieving a practical quantum computer. Since any information processing system is not a single device, it has to utilize many components, from high-level software packages to device-level libraries. In this study, we focus on one important subsystem, the arithmetic circuit. To design quantum arithmetic circuits, many researchers have exploited conventional classical arithmetic circuits such as the ripple-carry adder, the carry-lookahead adder, and the conditional sum adder [2, 3, 4, 5]. The quantum Fourier transform adder, in contrast, is genuinely based on quantum effects [6].

Unfortunately, most of these adders are based on some ideal model of a quantum computer which may not be physically realizable, such as assuming no limitation on interaction distance. Although studies with such optimistic assumptions will give us more confidence about the speedup of a quantum computer, they are still not precise enough to tell us the exact performance gain, and sometimes to difficult are match to a chosen architecture.

In this work, we focus on addition under more practical assumptions, such as a two-dimensional layout of qubits allowing nearest-neighbor interaction only, and supporting only two-qubit gates with concurrent execution, a model we call 2D NTC. Specifically, the two-dimensional layout of qubits can serve as a reasonable model for many technologies with fabrication, wiring, and control constraints. Because the input size of quantum gates is limited to one or two qubits, Toffoli gates must be broken down into smaller components. Surprisingly, for this architecture model, no specific quantum adder has been proposed. Van Meter and Oskin indicated that an adder would be $O(\sqrt{n})$ in time complexity on the 2D architecture, but no circuit was provided [7].

*bschoi3@gmail.com
†To whom any correspondence should be addressed.
‡rdv@sfc.wide.ad.jp

2 Summary of Contribution

In this work, we propose a new quantum adder on the 2D NTC architecture. More specifically, we contribute as follows:

- **Propose a 2D NTC adder.** We propose a 2D NTC adder, based on a combination of a ripple-carry adder and a carry-lookahead circuit, arranged in three phases. In the first phase, the first column executes a simple ripple carry addition and the other columns do carry lookahead operations to generate inter-column carry lookahead information. In the second phase, the carry output of the first column is propagated to the other columns sequentially to generate incoming carry for the corresponding column. In the last step, the incoming carry value for each column is propagated through the rows to generate carry values, and hence finally to generate summation values for each bit position. Once the third phase completes, the sum is copied out and the addition circuit is reversed to clean all ancillae.

- **Analyze storage requirements.** The necessary number of qubits for our adder is $5n - \sqrt{n} + 1$, inputs, ancillae, and copy out of results, as shown in Table 1. To work as concurrently as possible, our adder utilizes about twice the number of qubits compared to 1D NTC adders.

- **Analyze depth and compare with other adders.** The depth of other adders is recalculated based on the same physical constraints. The chosen adders for certain architectures and their depths are shown in Table 2. Based on the coefficients of each 1D NTC adder, our adder works faster when the input size is larger than 58.

- **Compare KQ values.** As discussed in Ref. [8], an important measure of the required resources is the product of the depth and size. Based on this analysis, we find when the input size is larger than 446, our adder is better than 1D NTC adders.

For more details of our adder and its analysis, please refer to the Appendix.

Table 1: Required Number of Qubits

Name	Number of qubits	Explanation
a_i	n	Input A
$b_i \rightarrow p_i \rightarrow s_i$	n	Input B, Carry propagate for i-th position, and Summation S
$\|0\rangle \rightarrow g_i \rightarrow G[i,j] \rightarrow c_i$	n	Carry generation for i-th position, Carry generation between i and j, and carry for i-th position
$\|0\rangle \rightarrow P[i,j]$	$n - 2\sqrt{n} + 1$	Carry propagation between i and j
$Column\_carry_k$	$\sqrt{n}$	Inter column carry. The last $Column\_carry$ is for the final carry output.
Sub Total	$(2n+1)+(2n-\sqrt{n})$	Mandatory + Additional
For Copy of Sum	n	
Total	$5n - \sqrt{n} + 1$	

Table 2: Depth and KQ, in Comparison with Other Adders

Architecture	Name of Adder	(Depth, Number of Qubits)	When is the present adder faster than the corresponding adder?	KQ[8]
1D NTC	VBE[9]	$(76n - 30,\ 3n + 1)$	$n > 2$	$228n^2 + \alpha_1$
	VBE-Improved[4]	$(20n - 15,\ 3n + 1)$	$n > 48$	$60n^2 + \alpha_2$
	CDKM[3]	$(18n + 14,\ 2n + 2)$	$n > 58$	$36n^2 + \alpha_3$
2D NTC	Present Adder	$(152\sqrt{n} - 104,\ 5n - \sqrt{n} + 1)$		$760n\sqrt{n} + \alpha_4$
AC	QFT-based[6]	$(3\log n,\ 2n + 1)$	N/A	$6n \log n$
	CLA-based[5]	$(2\log n + 2,\ 4n - \log n)$	N/A	$8n \log n$
	RCA+CLA-based[10]	$(10\log n + 6n/\log n,\ n + 4n/\log n)$	N/A	$10n \log n$

3 Future Work

Our new adder is the first design for a 2D NTC architecture, to the best of our knowledge, but we believe there is still room for improvement.

First, the number of gates added to the fundamental addition circuit is very large. Most of the gates are used for local shuffling of qubits to neighboring positions so that gates can be executed. This shuffling increases with the distance between qubits which have information dependency. Therefore, by arranging qubits in a better way, it may be possible to reduce the necessary number of operations.

Second, the phase for cleaning ancillae qubits doubles the number of quantum operations. In our adder, the ancillae qubits are re-initialized by applying the inverse circuit. Perhaps there would be a way to reduce such drawback by exploiting overlapping of the clearing phase with the computation phase.

Third, the number of ancillae qubits is also very large. Although our design uses additional ancillae in order to exploit parallel execution, this increases the depth as a negative effect and hence has to be minimized as well. The tradeoff between the depth and qubits may yet yield additional gains.

Acknowledgements

This research is supported by the Japan Society for the Promotion of Science (JSPS) through its "Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program)."

References

[1] Michele Mosca. Quantum algorithms. `http://arxiv.org/abs/0808.0369`, 2008.

[2] Yasuhiro Takahashi. Quantum arithmetic circuits: A survey. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E92.A(5):1276–1283, 2009.

[3] Steven A. Cuccaro, Thomas G. Draper, Samuel A. Kutin, and David Petrie Moulton. A new quantum ripple-carry addition circuit. `http://arxiv.org/abs/quant-ph/0410184`, 2004.

[4] Rodney Van Meter and Kohei M. Itoh. Fast quantum modular exponentiation. *Phys. Rev. A*, 71(5):052320, May 2005.

[5] Thomas G. Draper, Samuel A. Kutin, Eric M. Rains, and Krysta M. Svore. A logarithmic-depth quantum carry-lookahead adder. `http://arxiv.org/abs/quant-ph/0406142`, 2004.

[6] Thomas G. Draper. Addition on a quantum computer. `http://arxiv.org/abs/quant-ph/0008033`, 2000.

[7] Rodney Van Meter and Mark Oskin. Architectural implications of quantum computing technologies. *ACM Journal on Emerging Technologies in Computing Systems*, 2(1):31–63, 2006.

[8] Andrew M. Steane. Overhead and noise threshold of fault-tolerant quantum error correction. *Phys. Rev. A*, 68(4):042322, Oct 2003.

[9] Vlatko Vedral, Adriano Barenco, and Artur Ekert. Quantum networks for elementary arithmetic operations. *Phys. Rev. A*, 54(1):147–153, Jul 1996.

[10] Yoshinori Kawata, Satoshi Yayu, and Shuichi Ueno. An efficient quantum addition circuit : Extended abstract. *IEICE Technical Report. Circuits and systems*, 107(527):95–96, March 2008.

[11] Byung-Soo Choi and Rodney Van Meter. Effects of interaction distance on quantum addition circuits. `http://arxiv.org/abs/0809.4317`, 2008.

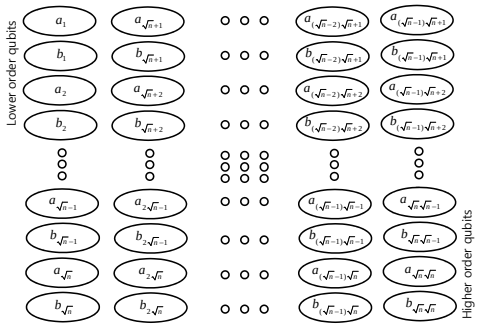


Figure 1: Layout of Input Qubits

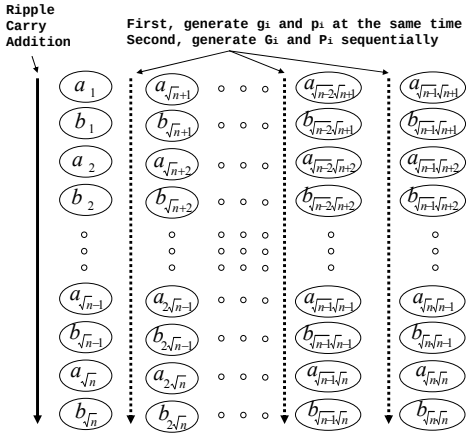


Figure 2: First Phase

Appendices

A Design

A.1 Layout of Qubits

The layout of qubits for our new adder on a 2D NTC architecture is shown in Figure 1. Two input registers $A = \sum_{i=1}^{n} 2^{i-1} a_i$ and $B = \sum_{i=1}^{n} 2^{i-1} b_i$ are interleaved in the square lattice structure. To create a compact layout, the size of rows and columns is $\sqrt{n}$ (for simplicity, we assume that $\sqrt{n}$ is an integer). Although other qubits are necessary for addition, which are used for intermediate storage for information dependency, they are not shown for simplicity. Since the overall depth heavily depends on the location of input qubits, the nearest neighbor layout of input qubits is best.

A.2 Three Phases

The adder consists of three phases: ripple carry addition and generation of carry lookahead information; generation of inter-column carry inputs; and generation of carry value for each position and summation.

In the first phase, as shown in Figure 2, the first column does ripple carry addition, and the other columns generate carry lookahead information for the corresponding column. Therefore, in the first column, the summation s_i for $i = 1 \ldots \sqrt{n}$ are generated, and the final carry output $Col\_carry_1 = c_{\sqrt{n}}$ is also generated, which will be used for the second phase. Meanwhile, carry lookahead information for the corresponding column can be extracted by generating $g_i = a_i \cdot b_i$ and $p_i = a_i \oplus b_i$ simultaneously, and

$$\begin{aligned} G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i] &= g_i \oplus p_i \cdot G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i-1], \\ P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i] &= p_i \cdot P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i-1] \end{aligned}$$

sequentially, where $i > \sqrt{n}$. Briefly, g_i indicates that the i-th position inputs *generates* carry output regardless of incoming carry, since both inputs are set to one. Likewise, p_i indicates that the i-th position inputs cannot generate carry output, but *propagate* incoming carry into carry output, since only one of inputs sets to one. $G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i]$ indicates that the subset of inputs from $(\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1)$ to i-th position generates carry output regardless of the incoming carry for the $(\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1)$-th position. $P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i]$ indicates that inputs from $(\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1)$ to i-th position cannot generate carry output, but propagate the incoming carry for $(\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1)$-th position. Note that for the first row of each column, i.e., when $i = k\sqrt{n} + 1$ for $k = 1, \ldots, \sqrt{n} - 1$, G and P are simply

$$\begin{aligned} G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, \lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1] &= g_i, \\ P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, \lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1] &= p_i. \end{aligned}$$

It is worth emphasizing that G and P for the last row for each column, $G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, (\lfloor i/\sqrt{n} \rfloor + 1)\sqrt{n}]$ and $P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, (\lfloor i/\sqrt{n} \rfloor + 1)\sqrt{n}]$, encompass all carry lookahead information for the corresponding column because of ripples of G and P between rows. Therefore, when the incoming carry value for the corresponding column is known, the corresponding column can decide the final carry out, which is the incoming carry value for the next column.

In the second phase, the final carry output of the first column $Col\_carry_1 = c_{\sqrt{n}}$ is propagated to the second column to generate the carry output Col_carry_2. As shown in Figure 3, the inter-column carry is evaluated by the following equation:

$$\begin{aligned} Col\_carry_j &= G[(j-1)\sqrt{n} + 1, j\sqrt{n}] \\ &\quad \oplus Col\_carry_{j-1} \cdot P[(j-1)\sqrt{n} + 1, j\sqrt{n}]. \end{aligned}$$

After this phase, all columns have outgoing carry value $Col\_carry_j = c_{j\sqrt{n}}$, where $j = 1, \ldots, \sqrt{n}$, which are used for incoming carry for the $(j+1)$-th column, respectively.

In the third phase, the incoming carry value for the corresponding column is propagated up each column, as shown in Figure 4, to generate the incoming carry value c_i for each bit position

$$\begin{aligned} c_i &= G[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i] \\ &\quad \oplus Col\_carry_{\lfloor i/\sqrt{n} \rfloor \sqrt{n}} \cdot P[\lfloor i/\sqrt{n} \rfloor \sqrt{n} + 1, i], \end{aligned}$$

where $i = k\sqrt{n}+2, \ldots, (k+1)\sqrt{n}$. The final carry output of the summation c_n is set to $Col\_carry_{\sqrt{n}}$. After this ripple, each position generates the summation value, $s_i = a_i \oplus b_i \oplus c_i$.

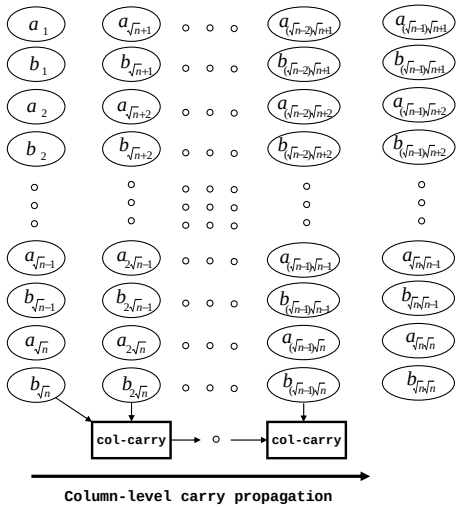


Figure 3: Second Phase

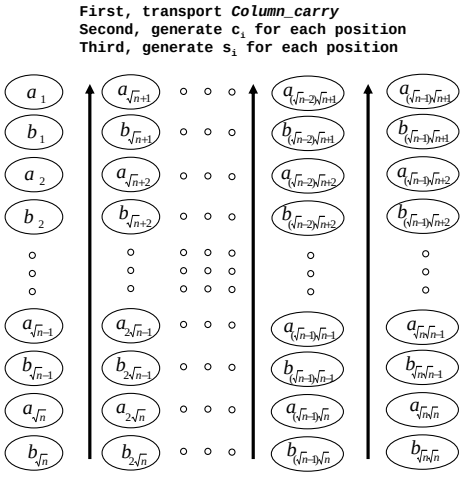


Figure 4: Third Phase

To clear the ancillae qubits, the circuit has to copy out all of the summation values into other qubits, and then apply the inverse of the addition circuit.

B Analysis

For fast computation, it is necessary to exploit parallelism as thoroughly as possible. Often, this requires the use of additional spatial resources. The storage requirements for our new adder are shown in Table 1. The additional number of qubits is $2n - \sqrt{n}$. For clearing ancillae, we need to copy the summation output to the additional qubits, adding n qubits. Therefore, the total number of qubits is $5n - \sqrt{n} + 1$.

When only interactions between neighboring qubits are allowed, the depth of an arithmetic circuit increases. For the 2D case, the depth lower bound was proved to be $\Omega(\sqrt{n})$ [11]. Our adder exhibits $O(\sqrt{n})$ depth since each phase needs $O(\sqrt{n})$ unit time steps. Therefore, our adder is asymptotically optimal for the 2D NTC architecture. However, it is necessary to compare to other adders designed for the 1D NTC architecture, since they can be implemented on the 2D NTC architecture without modification.

In the first phase of our adder, there are two flows, one for the first column and the other for other columns. For calculating the depth we assume only one-input and two input gates with unit time delay. The depth of the first column is the sum of the delay of one half adder and of $\sqrt{n} - 1$ full adders, for a total of $26\sqrt{n} - 16$. The other columns need a constant delay for generating g_i and p_i and $O(\sqrt{n} - 1)$ for generating $G[i, j]$ and $P[i, j]$, for a total of $36\sqrt{n} - 26$. Since the "long pole" determines the execution time, the depth for the first phase is $36\sqrt{n} - 26$. In the second phase, we have to execute $\sqrt{n} - 1$ column carry generation circuits, needing $18\sqrt{n} - 18$ delay. In the third phase, $\sqrt{n} - 1$ carry generation circuits must be executed sequentially, followed by the summation circuit. The delay for this is $21\sqrt{n} - 8$. Overall the depth of the circuit body is $75\sqrt{n} - 52$. Next, our adder has to copy the results, using another $\sqrt{n}$ swap operations, adding $3\sqrt{n}$ delay. Finally, it must undo the overall computation. Hence, the whole depth is $152\sqrt{n} - 104$.

The overall analysis and the comparison between the adders are shown in Table 2. The first column lists the architecture for adders, and the second column some adders for each architecture. For the 1D NTC architecture, we chose three typical adders. Vedral et al. proposed a plain adder based on the ripple carry [9], named *VBE* in the table. To improve the performance of *VBE*, Van Meter and Itoh proposed another version, *VBE-Improved* [4]. Cuccaro et al. proposed a ripple carry adder with only one ancillae qubit [3], named *CDKM*. For the 2D NTC architecture, our adder is shown. For the architecture with arbitrary distance interaction and concurrent execution (AC), several adders have been proposed. Draper proposed a quantum Fourier transform adder [6], named *QFT-based*. By exploiting the classical fast addition algorithm, Draper et al. also proposed a carry lookahead adder [5], named *CLA-based*. Kawata et al. also proposed an adder based on the combination of ripple carry adder and carry lookahead adder [10], named *RCA+CLA-based*.

For comparison, the depth and the size of each adder is shown in the third column. In this work, the depth is measured in units of one- and two-qubit gates for the 1D and 2D NTC architectures. The depth for AC architecture is based on one-, two-, and CCNOT gates. Note that numbers for AC architecture are not directly comparable because a Toffoli gate is assumed to cost only a single time step, and long-distance gates are allowed. In the fourth column, the input size is shown for which our adder works faster than the corresponding adder.

In the fifth column, we list the depth-storage product KQ, where K and Q are the numbers of computational steps and logical qubits, respectively [8]. KQ is a measure of the resources required for a particular circuit, and is especially useful for estimating the required strength of error correction. Successful execution of a circuit requires residual logical gate and memory error rate of $p_e << (KQ)^{-1}$. Note that the number of qubits includes input, output, and ancillae.

Algorithm Optimisation for Topological Measurement-Based Quantum Computing

Clare Horsman[1] * Simon Devitt[2] Rodney Van Meter[3]

[1] *Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, UK*
[2] *National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan.*
[3] *Faculty of Environment and Information Studies, Keio University 5322 Endo, Fujisawa, Kanagawa, Japan*

Abstract. We present methods for optimizing algorithm design in topological cluster state models of quantum computing. This is a crucial element in the design of large, scalable quantum architectures. We detail the resources required for defects to be propagated through the cluster to implement specific gates and subroutines, and give methods by which different types of resources (spatial and temporal) may be interchanged. We outline constructive methods for generating optimal defect patterns from a given circuit, significantly reducing the quantum resources required to implement key routines for quantum computing.

The development of a large-scale quantum computer is a highly sought-after goal for many groups worldwide. Recent theoretical developments have allowed for detailed research into an overall architectural structure for such a computer. The most comprehensive quantum computing designs are built fundamentally from the ideas of topological cluster states and surface codes [1,2,3].

These models for quantum computing are remarkably useful. Not only do they exhibit one of the largest fault-tolerance thresholds of any model (both for traditional qubit errors and also more pathological channels such as loss), but they are also flexible enough to incorporate into a large-scale architectural design. Such designs allow scaling to arbitrary size in a conceptually simple manner.

As these topological coding schemes are based on the concept of *measurement-based quantum computation*, the programming of such a device to perform quantum algorithms is a question of *classical* software control. The quantum hardware of such a computer is only responsible for the creation of a large, entangled cluster state. How this state is utilized to perform quantum algorithms and error correction protocols is dictated by measurements on the cluster. The classical control and optimization of such a computer is therefore an important question. This talk will briefly summarize the basic principles of implementing and optimizing the classical programming of a topological computer, in order to reduce the overall quantum resources for various subroutines and algorithms.

The concept of algorithm optimization for the topological cluster model can be easily defined. Figure 1a) shows the standard braiding pattern to perform a CNOT operation between logically encoded defect qubits defined within a 3D cluster. Figure 1b) illustrates a sequence of CNOT gates, performed during a large-scale quantum algorithm. When implementing a standard quantum subroutine (such as arithmetic gates, quantum Fourier transforms, oracle queries, etc.) circuit decompositions are converted into a sequence of braiding operations. In addition, each basic quantum gate must be decomposed in terms of the primitive operations which are valid within the topological model. These primitive operations include specific state initialization, measurement and CNOT operations. More complicated gates require the introduction of singular states, magic state distillation protocols, and teleported gates; these operations also require decomposition into a basic braiding pattern.

In order to optimize the braiding sequence for quantum subroutines, we wish to compact a braiding pattern into the smallest possible cluster volume (given the error correction requirements of the quantum code). The error correction strength can be defined by the distance of the underlying quantum code, d, and is determined by both the fundamental physical error rates in the computer and the size of the specific algorithm. Once this distance is specified, defects are defined in the lattice which have a circumference of d and are separated from each other also by d. Figure 1a) shows an optimized CNOT gate for a distance $d = 8$ topological code. For this braiding operation, the total volume of the cluster is such that all defect regions are separated from each other by the minimum possible amount (in this case, 8 cluster cells).

When constructing a quantum subroutine, we naively decompose the circuit into its primitive components (CNOT, Toffoli, Hadamard etc.) and then define a braid and defect pattern to be implemented in the computer. As gates such as the Toffoli cannot be implemented directly on the lattice and instead require the preparation of distilled ancilla states and teleportation protocols, directly constructing a braiding sequence for a large subroutine will not in general lead to an optimized pattern.

The purpose of this project is to construct a braiding sequence for a large quantum subroutine (such as a quantum adder or quantum Fourier transform), and then develop general techniques to minimize the total cluster volume needed to implement the braid pattern. The ul-

*clare.horsman@bristol.ac.uk

timate goal is to achieve a optimal braiding sequence in a minimal volume, where all defect separations in the cluster are d, and the cluster prepared by the quantum hardware contains no wasted space.

When considering the possible patterns of defects and braids for a gate sequence, we must first take into account the constraints that the code distance puts on how defects can move. These layout restrictions can be completely characterised for single qubit/braid defects and also for the interaction between two or more defect pairs when they intersect. The characterisation includes the resources required to complete a particular layout move; for some layouts spatial and temporal resources may be interchanged to a certain extent, but for others a minimum amount of each may be required. One example is when a single defect pair is propagated vertically within a 2-dimensional timeslice (figure 2). In order to preserve the code distance, the operation requires either an additional parallel row of cells in the cluster state (figure 2a), or else a second timeslice surface (figure 2b). So a choice may be made between spatial or temporal costs in that case. By contrast, when two defects intersect (figure 3) there is an irreducible temporal cost: one must propagate over the top of the other temporally, requiring an extra 'timeslice' (2-dimensional spatial surface) to complete the movement.

One particularly important operation is swapping the position of two qubits – that is, interchanging two defect pairs. With careful manipulation of the four defects, it is possible to perform this operation with only one additional cluster cell, although this then requires six different timeslice surfaces to complete (figure 4). This then enables us to give resource requirements for all algorithms in the extreme cases where either temporal or spatial resources are required to be minimal, but the other resource type is unbounded. With no restrictions on temporal resources, the minimum spatial resource for any algorithm is one additional cell. For an unbounded amount of spatial resource, any sequence of gates with a computational depth of 1 may be performed with only one additional timeslice.

When performing two-qubit gates, in general we have two options. Either we can propagate long-distance braids between the qubits, or we can move the qubits and then perform shorter-range braiding. If we consider the motion of the braids and the defects together, we obtain the concept of a *path* between qubit defects. This path can be realised by a combination of qubit defect movement and/or braiding. An algorithm is then defined by a series of paths through the cluster, and optimisation concerns minimising the cluster volume they require.

If a gate sequence can be laid out on a 2-dimensional timeslice such that there exist paths that neither intersect nor overlap, then the sequence can be performed in a single timestep. However, if no such set of paths exist, then additional timeslices are needed. For two paths to cross, at least one extra time surface is required. More surfaces may be needed depending on whether the path is instantiated as qubit or braid defect movement. If at least one of the paths is a qubit movement, then the requirement that qubits must persist through time may mean that more than one extra surface is needed at an intersection. However, if two braid defects cross then they always require only a single extra surface. This is furthermore also the case when any number of paths cross: an intersection of n paths requires at least n different time surfaces to perform.

We now have a simple way of finding the minimum time for an algorithm given a certain amount of space. Laying the qubit defects out on a 2-dimensional surface, we find the paths between them corresponding to the required gates such that the maximum number of intersecting paths is minimised. That is then the minimum time required. Moreover, these paths give the construction of the minimum-time algorithm itself: leaving the qubits static, they are the paths along which long-range braids are to be propagated. It is then easy to give provably minimum-time constructions by hand of simple algorithms such as a mesh circuit. Once paths have been found between the qubits, the next step is to operate on the path collection to "squeeze" it so that each path then becomes contiguous with its neighbours, and there is no space on the cluster between paths. It is currently ongoing work to prove the conjecture that this squeezing operation preserves the topology of the defect structure, and that all defect configurations in the same homotopy class perform the same unitary evolution.

This work has many implications for topological measurement-based computing. By reducing to a minimum the quantum resources required to implement certain subroutines, it helps decrease the technological barriers to quantum computing. Giving detailed methods for implementing these optimal routines enables the creation of efficient automated compilers for the topological model, translating logical gate sequences into physical operations on the system. Furthermore, these results lay the groundwork for large-scale algorithm design that works natively in the topological model, allowing the efficient programming of a large-scale multi-task quantum computer.

References

[1] R. Raussendorf, J. Harrington, K. Goyal. Topological fault-tolerance in cluster state quantum computation. New J. Phys. 9, 199 (2007).

[2] S. J. Devitt et al. Architectural design for a topological cluster state quantum computer. New. J. Phys. 11, 083032 (2009).

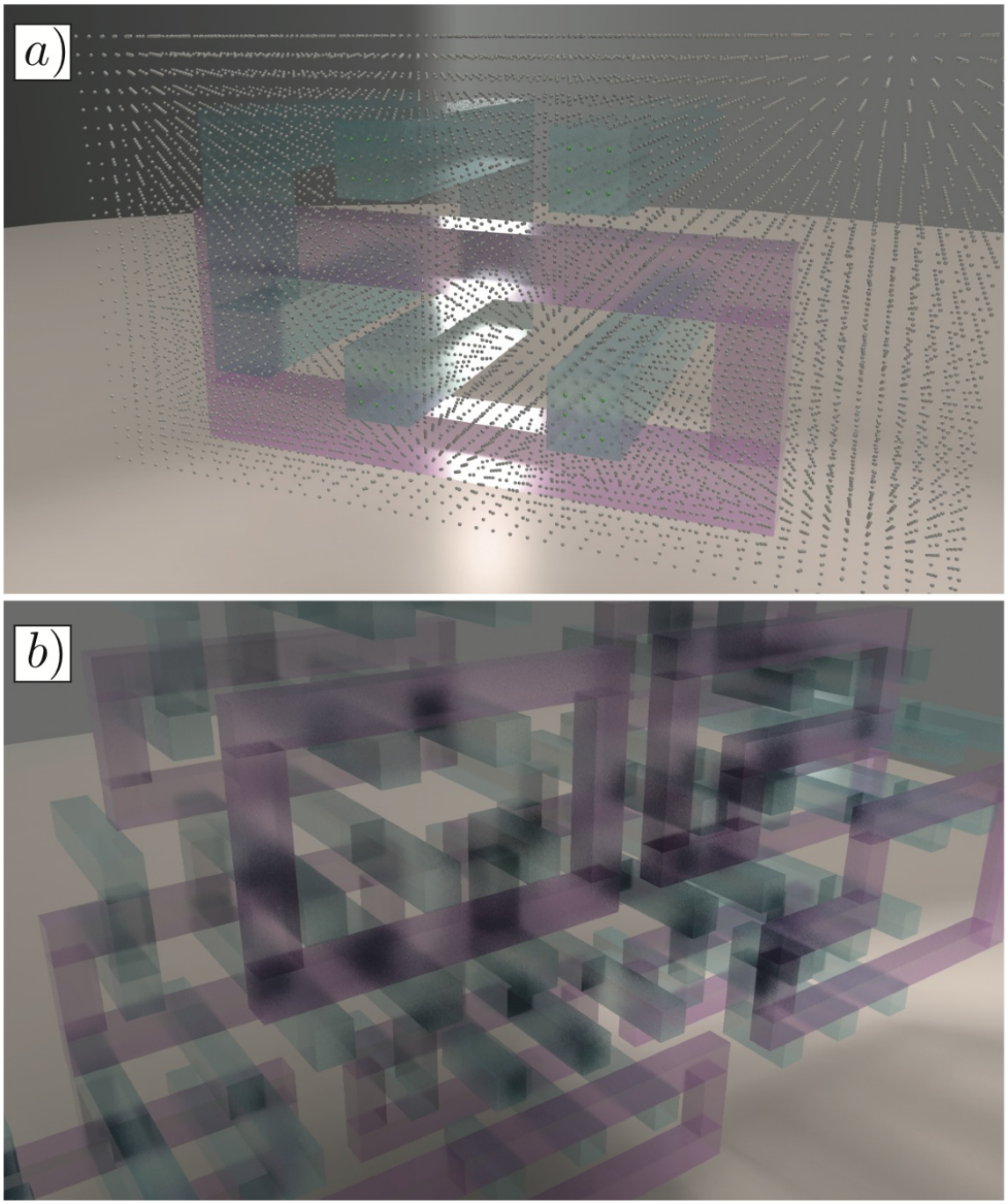
[3] A. G. Fowler et al. High threshold universal quantum computation on the surface code. Phys. Rev. A 80, 052312 (2009).

[4] R. Van Meter et al. Distributed quantum computiataion architecture using semiconductor nanophotonics. International Journal of Quantum Information 8, 295–323 (2010)

Acknowledgements

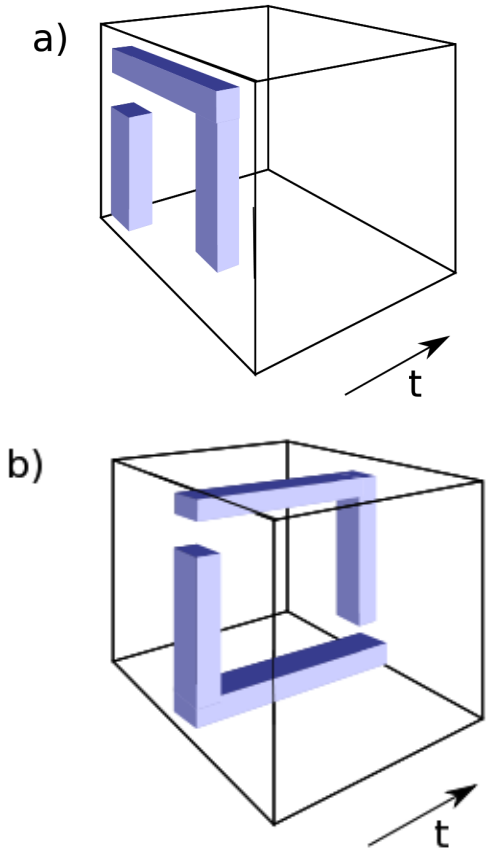
This research is supported by the Japan Society for the Promotion of Science (JSPS) through its "Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program)."

Figure 1



Braiding patterns for a) a single CNOT gate and b) multiple CNOT gates. The dots represent physical qubits, and the different coloured regions represent different types of defect, for example the blue regions are computational qubits. Braiding operations can only take place between defects of different types. The introduction of an ancilla (purple) qubit allows us to braid two qubits of the same (blue) type.

Figure 2



Propagating a defect pair vertically, using a) extra spatial resources and b) extra temporal resources.

A Layered Architecture for Quantum Computing using Optically-Controlled Quantum Dots

Cody Jones[1,*], Austin G. Fowler[2], Jungsang Kim[3], Thaddeus D. Ladd[1,4,†], Rodney Van Meter[5], and Yoshihisa Yamamoto[1,4]

[1]*Edward L. Ginzton Laboratory, Stanford University, Stanford, California, USA*
[2]*Centre for Quantum Computing Technology, University of Melbourne, Victoria, Australia*
[3]*Fitzpatrick Institute for Photonics, Duke University, Durham, NC, USA*
[4]*National Institute of Informatics, Japan*
[5]*Faculty of Environment and Information Studies, Keio University, Japan*
[*]*Corresponding author: ncjones@stanford.edu*

Abstract. Designing a quantum computer capable of executing large-scale programs such as Shor's algorithm is a challenging engineering problem. We develop a complete framework for a quantum computer based on optical control of quantum dots in a layered architecture, showing how a variety of technologies must operate synchronously in such a system. Furthermore, we investigate the performance and manufacturability of this system. We find that Shor's factoring algorithm for a 2048-bit number can be executed in approximately 14.4 days. Moreover, this architecture can leverage many existing technologies from mature industries, making the development of a functioning quantum computer a feasible endeavor in the near future.

1 Introduction

We are designing a quantum computer, with the principal imperative to take advantage of the strong capabilities of device integration afforded by semiconductor fabrication. Our qubit is defined by the electron spin states of a charged quantum dot controlled by ultrafast optical pulses [1, 2]. Optical control makes this system very fast, scalable to large problem sizes, and extensible to quantum communication [3, 4] or distributed architectures [5]. The design of this quantum computer centers on error correction in the form of a topological surface code, which requires only local and nearest-neighbor gates. The framework of this architecture is flexible, permitting analysis and comparison of other quantum computer designs.

2 Layered Framework

The present architecture is a control stack consisting of four layers, where each layer has a prescribed set of duties to accomplish. As shown in Figure 1, the interface between two layers is defined by the services a lower layer provides to the one above it. To execute an operation, a layer must issue commands to the layer below and process the results. The various functions in a quantum computer are organized in a manner which aids understanding for the designer and translates directly into an effective control structure for the device itself. This approach facilitates isolation of design problems in individual layers, wholesale replacement of an entire layer with a different technology, and independent evolution of layers over time.

At the top of the control stack is the logical layer, where a quantum algorithm is implemented and results are provided to the user. The bottom hosts the raw physical processes underpinning the quantum computer. The layers between (virtualization and surface code) are essential for shaping the imperfect quantum processes into a system of high-accuracy quantum gates at the logical layer.

2.1 Layer 1: Physical

The lowest layer houses the specific hardware and physical processes. In simplest terms, this is the storage and manipulation of unprotected quantum information. The physical layer provides the essential physical processes to satisfy the virtualization layer. The system we propose uses charged quantum dots to store quantum information in the electron spin state [6, 7, 8, 9, 10, 11]. The quantum dots are embedded in a planar distributed Bragg reflector (DBR) microcavity [12], and a transverse magnetic field separates the spin levels [13]. Broadband optical pulses rotate the spin vector [1, 2]; these optical pulses are controlled by arrays of MEMS micromirrors [14, 15]. Dispersive readout provides measurement of the spin state [16, 17].

2.2 Layer 2: Virtualization

The virtualization layer is where quantum systems are organized into information units and operations. Decoherence processes such as environmental noise and control errors mandate that an intermediary processing step transforms the raw subsystems in the physical layer into "virtual qubits" and "virtual gates" for the surface code

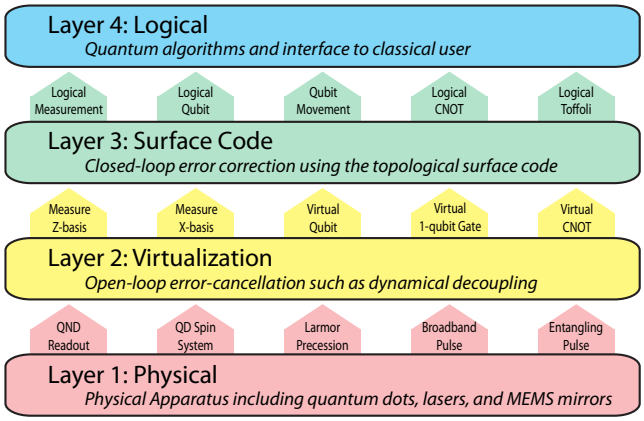


Figure 1: Layered control stack which forms the framework of a quantum computer architecture. Arrows indicate services provided to a higher layer.

layer. The virtualization layer is distinct from the surface code layer in that Layer 2 contains only open-loop control; no readout or active feedback is implemented.

A Virtual Qubit is a quantum two-level system whose state is protected by dynamical decoupling (DD), decoherence-free subspaces (DFS) [18], or some other manner of open-loop control. In this particular system, static decoupling sequences protect the virtual qubit from dephasing by the nuclear environment [19].

A virtual gate is an operation on a virtual qubit to change its state, typically with reduced error from the raw processes in the physical layer. This may include composite physical gate sequences with net suppressed error. This architecture embeds the BB1 compensation sequence [20] within the static decoupling sequence so that the ensemble corrects both dephasing and pulse-induced errors.

2.3 Layer 3: Surface Code

The surface code layer provides the ability to correct arbitrary errors with quantum error correction [21, 22]. In contrast to Layer 2, the surface code is a closed-loop system used to periodically measure an error syndrome and correcting errors. The probability of an undetected error decreases exponentially as a function of the "distance" of the code, so that logical qubits and gates with arbitrarily low error can be produced with a sufficiently large code. However, the virtual qubits and gates must have error rates below the threshold of the surface code (0.75%), so that often Layer 2 techniques are necessary for Layer 3 to function. The surface code layer uses closed-loop error correction to provide fault-tolerant logical qubits, logical gates, and logical measurement to Layer 4. The primary control loop of this architecture appears in Figure 2.

In addition to a relatively high threshold, the surface code also tolerates delayed feedback from measurement of the error syndrome [23, 24]. Logical operation speed (and hence algorithm runtime) may suffer, but the code strength is preserved. This feature permits Layer 3 to be robust against the unpredictable error correction stage — the error syndrome must be translated into a most-probable logical error, a process which can be computationally intensive for a large surface code.

2.4 Layer 4: Logical

The logical layer hosts the quantum algorithm that a classical user wishes to execute. Operations are performed on the logical qubits provided by the surface code, and the end result is communicated to the classical user. The system designed here executes Shor's algorithm [25] to factor a 2048-bit number. As indicated in Figure 3, the speed of logical operations depends on many performance aspects of Layers 1, 2, and 3, such as optical pulse repetition rate and bandwidth, fidelity of virtual gates after dynamical decoupling sequences, and the size of the surface code needed for the chosen algorithm.

3 Prospects

In this investigation we seek to design a quantum computer architecture using optically-controlled quantum dots which can be scaled to problems believed to be intractable for classical computation, such as Shor's algorithm. We estimate this can factor a 2048-bit number in 14.4 days by taking advantage of ultrafast optical control. Additionally, this particular architecture attempts to take advantage of mature technologies from other fields wherever possible, such as the MEMS mirror arrays. We implement the surface code for quantum error correction; this has a strong influence on the physical design of this system (such as nearest-neighbor gates for a 2D array of quantum dots), but in principle other methodologies like CSS codes [26] are compatible with this framework.

What hardware performance is necessary to build a functional quantum computer? Common figures of merit are fidelity, operation time, and qubit coherence time. Here we go further to show how connectivity and classical control performance are also crucial. To design a quantum computer requires viewing the system as a whole, such that tradeoffs and compatibility between component choices may be addressed. We explore how to approach the complete challenge of designing a quantum computer. Furthermore, our results present a template for developing quantum architectures based on other technologies, so that differing approaches to quantum information, such as ion traps or optical lattices, can be contrasted in a meaningful context. In the future we intend to explore more rigorously the specific hardware requirements of this system, such as developing the integrated circuit required for error syndrome processing in Layer 3, or the optical projection system for laser pulses in Layer 1.

4 Acknowledgments

This work was supported by the National Science Foundation CCF-0829694, the Univ. of Tokyo Special Coordination Funds for Promoting Science and Technology, NICT, and the Japan Society for the Promo-

tion of Science (JSPS) through its "Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program)." CJ was supported by the National Science Foundation Graduate Fellowship.

References

[1] Press D, Ladd T D, Zhang B and Yamamoto Y 2008 *Nature* **456** 218–221

[2] Berezovsky J, Mikkelsen M H, Stoltz N G, Coldren L A and Awschalom D D 2008 *Science* **320** 349–352

[3] Briegel H J, Dür W, Cirac J I and Zoller P 1998 *Physical Review Letters* **81** 5932–5935

[4] Dür W, Briegel H J, Cirac J I and Zoller P 1999 *Physical Review A* **59** 169–181

[5] Van Meter R, Ladd T D, Fowler A G and Yamamoto Y 2010 *International Journal of Quantum Information* **8** 295–323 preprint available as arXiv:0906.2686v2 [quant-ph].

[6] Björk G, Pau S, Jacobson J and Yamamoto Y 1994 *Physical Review B* **50**

[7] Imamoglu A, Awschalom D D, Burkard G, DiVincenzo D P, Loss D, Sherwin M and Small A 1999 *Physical Review Letters* **83** 4204–4207

[8] Bonadeo N H, Chen G, Gammon D and Steel D G 2000 *Physica Status Solidi B* **221** 5–18

[9] Guest J R, Stievater T H, Li X, Cheng J, Steel D G, Gammon D, Katzer D S, Park D, Ell C, Thränhardt A, Khitrova G and Gibbs H M 2002 *Physical Review B* **65**

[10] Hours J, Senellart P, Peter E, Cavanna A and Bloch J 2005 *Physical Review B* **71**

[11] Yamamoto Y, Ladd T D, Press D, Clark S, Sanaka K, Santori C, Fattal D, Fu K M, Höfling S, Reitzenstein S and Forchel A 2009 *Physica Scripta* **T137**

[12] Reitzenstein S, Hofmann C, Gorbunov A, Strau M, Kwon S H, Schneider C, Lffler A, Höfling S, Kamp M and Forchel A 2007 *Applied Physics Letters* **90**

[13] Bayer M, Ortner G, Stern O, Kuther A, Gorbunov A A, Forchel A, Hawrylak P, Fafard S, Hinzer K, Reinecke T L, Walck S N, Reithmaier J P, Klopf F and Schäfer F 2002 *Physical Review B* **65**

[14] Kim C, Knoernschild C, Liu B and Kim J 2007 *IEEE Journal of Selected Topics in Quantum Electronics* **13** 322–329

[15] Knoernschild C, Kim C, Liu B, Lu F P and Kim J 2008 *Optics Letters* **33** 273–275

[16] Berezovsky J, Mikkelsen M H, Gywat O, Stoltz N G, Coldren L A and Awschalom D D 2006 *Science* **314** 1916–1920

[17] Atatüre M, Dreiser J, Badolato A and Imamoglu A 2007 *Nature Physics* **3** 101–106

[18] Bacon D 2001 *Decoherence, control, and symmetry in quantum computers* Ph.D. thesis University of California, Berkeley

[19] Press D, Greve K D, McMahon P L, Ladd T D, Friess B, Schneider C, Kamp M, Höfling S, Forchel A and Yamamoto Y 2010 *Nature Photonics* **4** 367–370

[20] Brown K R, Harrow A W and Chuang I L 2004 *Physical Review A* **70**

[21] Raussendorf R, Harrington J and Goyal K 2007 *New Journal of Physics* **9**

[22] Raussendorf R and Harrington J 2007 *Physical Review Letters* **98**

[23] Devitt S J, Fowler A G, Stephens A M, Greentree A D, Hollenberg L C L, Munro W J and Nemoto K 2009 *New Journal of Physics* **11**

[24] Fowler A G, Stephens A M and Groszkowski P 2009 *Physical Review A* **80**

[25] Shor P W 1997 *SIAM J. Comput.* **26** 1484–1509

[26] Nielsen M A and Chuang I L 2000 *Quantum Computation and Quantum Information* 1st ed (Cambridge University Press) ISBN 0521635039

†Present address: HRL Laboratories, LLC, 3011 Malibu Canyon Rd., Malibu, CA 90265

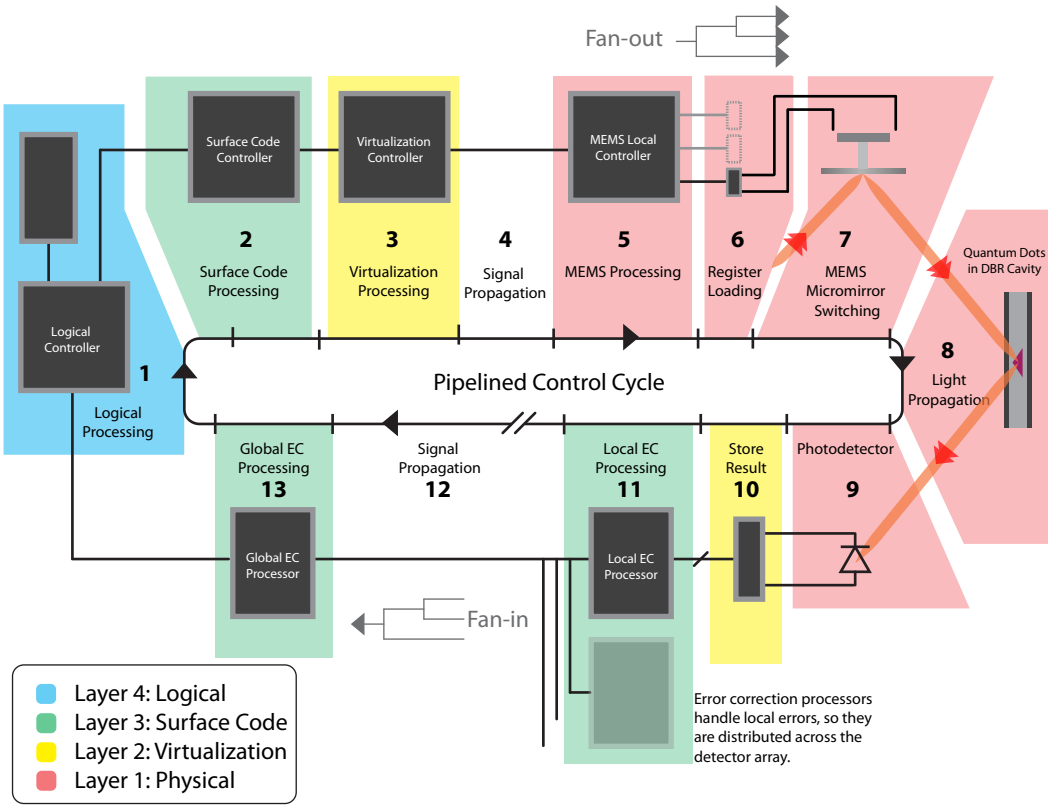


Figure 2: Primary control cycle of the quantum computer. Though presented as a sequence of steps, each layer issues multiple commands to the layer below, and operations are pipelined. The true quantum behavior exists in step 8, where ultrafast laser pulses interact with charged quantum dots, and in step 9, where the qubit state is measured by detecting the polarization of light reflected from the DBR cavity containing the quantum dots.

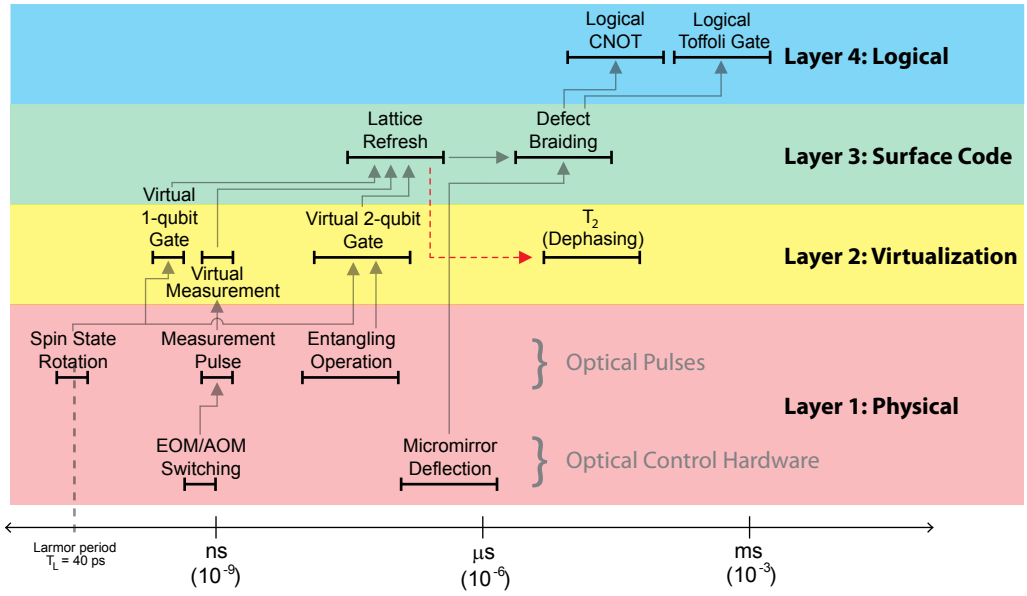


Figure 3: Relative timescales for critical operations within each layer. The arrows indicate dependence of higher operations on lower layers. The red arrow signifies that the surface code lattice refresh must be much faster than the dephasing time in order for error correction to function.

Defective Qubits in Surface Code Quantum Computation on a Fixed Lattice

Shota Nagayama[1] * Rodney Van Meter[2] †

[1] *Graduate School of Media and Governance, Keio University*
5322, Endo, Fujisawa City, Kanagawa, Japan.
[2] *Faculty of Environment and Information Studies, Keio University*
5322, Endo, Fujisawa City, Kanagawa, Japan.

Abstract. Surface code quantum computation is a feasible architecture for fault-tolerant quantum computation. However, there are some physical problems that must be overcome to implement it. With solid-state systems, physically non-functional devices may exist, affecting our ability to correct errors. We describe the relationship between the yield and physical and logical error rates, using realistic models for the error rate for both plaquette and superplaquette operations.

Keywords: Surface code quantum computing, yield, error rate

1 Introduction

Surface code quantum computation[1] is a feasible architecture for fault-tolerant quantum computation, but it has several problems to be solved. One such problem is how to deal with physical defects. If the number of defects is large, we need a large lattice, losing efficiency of the use of physical resources. Stace, Barrett and Doherty[2] addressed the loss problem in an idealized environment. They assumed the ability to measure "superplaquette" operators perfectly, an assumption that raises the threshold by an order of magnitude. They do not consider hole movement, which is critical. Operations in the surface code are done by moving two holes in the lattice. Physical error chains which connect holes result in logical errors, so the distance between holes determines the tolerance property. Hole movements will change this distance. Our in-progress simulations describe the tolerance properties of a defective lattice. Our goal is to determine the minimum lattice hole size and spacing, given the yield and physical gate error rate, for non-idealized conditions.

2 Surface code quantum computing

In surface code quantum computing, a lattice consisting of physical qubits represents one or more logical qubits. The correlation between a pair of lattice boundaries represents a logical qubit. We concentrate here on surface code quantum computing based on solid state physics, for example quantum dots with unpaired electron spins as the qubits, and cavity QED for control.

We use the term "hole" where Raussendorf et al.[3] and Fowler et al. use "defect". A hole is a region of the lattice that is not used. The boundary of a hole gives a degree of freedom that can be used as a logical qubit. We use the term "defect" to refer to physical qubit errors such as non-functional quantum dots. Note that a defect may act as an immobile hole, which may either disturb the computation or be used in computation.

The yield is the ratio of quantum dots of sufficient quality to the total number of dots produced. Yield is affected by spectral inhomogeneity, poor cavity quality, and other factors, depending on quantum dot type. The lower the yield, the bigger the lattice size should be to achieve the same error correction strength.

The physical error rate (P_{phy}) is the probability that a physical gate or other physical error occurs. Physical errors are detected with stabilizer measurements. We call a set of qubits which consist of a syndrome qubit and 4 data qubits a "plaquette". Measurements on this unit give the stabilizer measurements of the plaquette. A "superplaquette" is a over-sized plaquette, formed around one or more defects. A defect connects two plaquettes around it into a superplaquette. Two defects may connect three plaquettes. We need different sequences of measurements to measure the stabilizers of different superplaquettes. The adjusted error rate (P_{adj}) is the probability of an error chain crossing a plaquette or superplaquette, adjusted for the probable errors on measuring the stabilizer. The logical error rate depends on the distance between two holes or between a boundary of the lattice and a hole. The shortest distance determines the logical error rate $P_{log} = (P_{adj})^{distance}$.

3 Superplaquettes

As noted above, there may be various types of superplaquettes, consisting of various numbers of single plaquettes, chained in various shapes. Each type of superplaquette has a specific equation to calculate its P_{adj}, see Table 1 and Figure 1, based on the number of gates needed to implement the superplaquette operator. The relationship between P_{adj} and the number of gates is $P_{adj} = 1 - (1 - P_{phy})^{\#gates}$. Optimizing the stabilizer circuit for arbitrary superplaquettes is an open problem that we think can be solved using graph theory, see Appendix A.2.

4 The influence of yield on logical error rate

Figure 2 shows the effect of low yield on the surface code. The presence of defects treated as superplaquettes shortens error chains between holes. The blue error chain requires fifteen lattice qubit bit flips, while the red

*kurosagi@sfc.wide.ad.jp
†rdv@sfc.wide.ad.jp

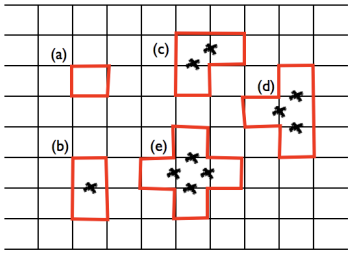


Figure 1: Plaquettes and superplaquettes. "×" denotes a defect. (a) Defect-free plaquette. (b) A 2-superplaquette formed by a single defect. (c)-(e) Some 3- to 5-superplaquettes formed by multiple defects.

case	#gates	P_{adj}
(a)	5	$P_{adj:(a)} = 1-(1-P_{phy})^5$
(b)	12	$P_{adj:(b)} = 1-(1-P_{phy})^{12}$
(c)	15	$P_{adj:(c)} = 1-(1-P_{phy})^{15}$
(d)	15	$P_{adj:(d)} = 1-(1-P_{phy})^{15}$
(e)	19	$P_{adj:(e)} = 1-(1-P_{phy})^{19}$

Table 1: The number of physical gates needed to implement the stabilizer operation of each superplaquette in Figure 1, and the corresponding P_{adj}.

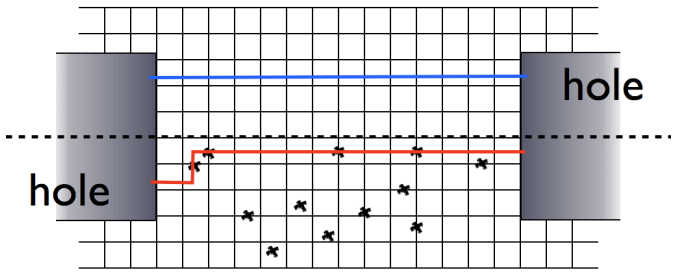


Figure 2: Examples of error chains. This blue line represents the ideal situation. No defects, or superplaquettes reduces the tolerance property. However, the situation around the red line is more realistic with superplaquettes and error chains that can run over them. Superplaquettes reduce the tolerance property, so the logical error rate will increase.

chain requires only eleven. The logical error rate of the blue error chain is $P_{log} = (P_{adj:(a)})^{14}$. This equation describes only the probability of a logical error occuring, not the probability of measuring the -1 eigenstate either correctly or incorrectly. In this situation there are defects in the lattice, so the shortest error chain will be the red line. It passes across only eleven qubits, so the error equation will be $P_{log} = (P_{adj:(a)})^7 \times (P_{adj:(b)})^3 \times (P_{adj:(c)})$. Superplaquettes act like a "bypass" around two or more plaquettes, reducing the tolerance property directly.

The logical error rate depends mainly on the shortest path of the error chains. We are developing computer simulations to reveal the relationship between yield and the shortest path in the channel between two holes. Dijkstra's algorithm is used to find the shortest path.

5 Summary

We have shown that the presence of defects significantly reduces the error correction capability of the surface code. Not only are defects in themselves computationally useless, but they also reduce other qubits' efficiency. We suggest the following as open problems:

1. **Superplaquette overhead** Superplaquettes need stabilizer circuits specific to their individual shape. The memory error rate of non-defective qubits may increase if they have to wait longer for superplaquette operations to complete.

2. **Full path simulation** Our initial simulations do not enumerate all patterns of paths. While the logical error rate increases exponentially with the path length, errors on non-shortest paths may affect the overall logical error rate. If there are many paths whose error rate is not small enough to be ignored, the result of our simulation may be unduly optimistic.

3. **Defective syndrome qubit** We did not assume that syndrome qubits can be defects. Defective syndrome qubits will affect the stabilizer operation significantly.

Once we have P_{adj} for every plaquette and superplaquette, we can build a weighted graph. Using the logarithm of every P_{adj} as the edge weight then allows us to apply Dijkstra's algorithm to determine the single most probable error path. For high defect rates, we expect this path to dominate the overall error probability. For nearly homogeneous lattices with few defects, large numbers of paths with similar probabilities will exist, requiring more careful counting and/or simulation to determine P_{log}.

Acknowledgements

This research is supported by the Japan Society for the Promotion of Science (JSPS) through its "Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program)." We would also like to thank Austin G. Fowler, Nathan Cody Jones, Simon Devitt, Byung-Soo Choi and especially Clare Horsman.

References

[1] Austin G. Fowler, Ashley M. Stephens, and Peter Groszkowski. High-threshold universal quantum computation on the surface code. *Phys. Rev. A*, 80(5):052312, Nov 2009.

[2] Thomas M. Stace, Sean D. Barrett, and Andrew C. Doherty. Thresholds for topological codes in the presence of loss. *Physical Review Letters*, 102(20):200501, 2009.

[3] Robert Raussendorf, Jim Harrington, and Kovid Goyal. Topological fault-tolerance in cluster state quantum computation. *New Journal of Physics*, 9:199, 2007.

A Appendix

A.1 Lattice design

Figure 3 shows the design of a lattice. Each hole is made to be $(d \times d)$, so it has $4d$ circumference. The distances between holes and between each hole and boundaries of the lattice are at least $4d$. Thus, when the yield is 1, which is the ideal situation, we can take the power "distance" as $4d$.

$$P_{log} = (P_{adj})^{4d} \tag{1}$$

$$d = f(yield, P_{adj}) \tag{2}$$

A.2 Graph problem in superplaquette operation

Superplaquette operations are shaped as chains of CNOT gates. Figure 4 shows an example of a 3-superplaquette in a straight chain. Data/syndrome qubits are nodes, and edges are between adjacent qubits in the graph determining the number of gates. We find that the diameter of the graph determines the number of gates.

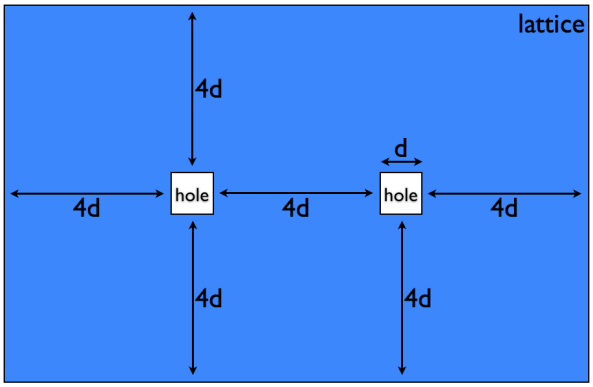


Figure 3: Lattice model.

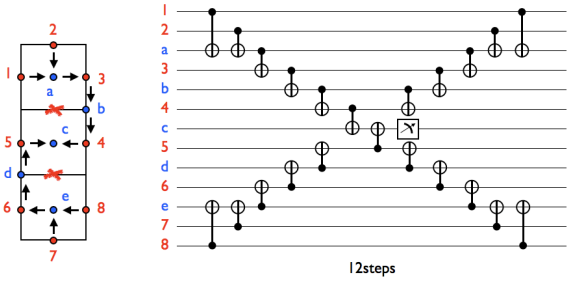


Figure 4: Circuit of an operation for a 3-superplaquette. This operation needs 22 gates and 1 measurement, 12 time steps.

Heterogeneous Interconnects in a Semiconductor Nanophotonic Surface Code Quantum Computer

Rodney Van Meter[1], Thaddeus D. Ladd[2,4,†], Austin G. Fowler[3], and Yoshihisa Yamamoto[2,4]

[1] Faculty of Environment and Information Studies, Keio University, 5322 Endo, Fujisawa, Kanagawa, 252-0882, Japan
[2] Edward L. Ginzton Laboratory, Stanford University, Stanford, CA, 94305-4088, USA
[3] Center for Quantum Computing Technology, University of Melbourne, Victoria 3010, Australia
[4] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

Abstract. The problem of communication will dominate both architectural choices and application performance in large-scale quantum computers. We propose a system based on semiconductor nanophotonics and electron spins in quantum dots that utilizes different communication protocols depending on the expected fidelity of connection. Although the surface code requires, at minimum, only two-dimensional nearest-neighbor coupling, our multi-level approach solves problems with heterogeneous coupling created by device layout limitations and by defective qubits, and supports inter-chip communication in the same framework.

Keywords: nanophotonics, surface code, quantum computer architecture

1 Introduction

Few of the many proposals for quantum computer architectures take an explicitly heterogeneous approach to designing systems, but the limitations of device size, classical control hardware, and quantum interconnects make perfectly uniform, large-scale systems impossible. As quantum architectures mature, they will adopt multi-level interconnect systems. The most visible distinction is between coupling within one chip or device, and coupling between chips, though intra-chip connections may also come in different flavors. Connectivity requirements strongly depend on the means chosen for quantum error correction.

Surface code quantum computation has one of the highest error thresholds for systems with constrained interconnects, at around 0.75% for gate and memory errors [4]. The logical error rate improves rapidly as the distance of the code is increased, and the resources required can be scaled incrementally. Moreover, movement of logical qubits or execution of gates between distant logical qubits can be done in nearly constant time, at the expense of space, allowing time performance of some algorithms (such as arithmetic) to be asymptotically optimal.

The surface code requires qubits arranged in a two-dimensional lattice. With the prospect of a low yield of qubits that meet our quality criteria, and a system architecture that does not support direct physical coupling to neighbors in all four cardinal directions, our proposed system couples qubits through heterogeneous communication channels that pass through different numbers, types and sizes of optical components. This requires the use of purification [1, 2] as well as different physical gate mechanisms.

In this paper, we present a nanophotonic architecture using the surface code. First, we describe the core components with a focus on the heterogeneity of the interconnect, followed by a discussion of the expected performance of very large-scale systems, some of the lessons learned, and the resulting demands on the evolution of the underlying technology.

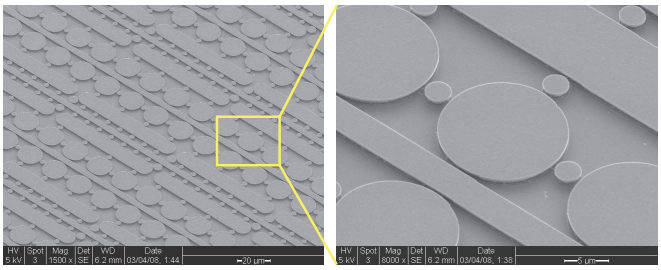


Figure 1: Scanning electron micrograph of a non-functional GaAs demonstration device with (unshown) InAs quantum dot layer, created to explore fabrication issues.

2 Architecture

We are developing an architecture that utilizes unpaired electron spins held in quantum dots in optical cavities [5]. Coupling is performed using different physical gates, dependent on the loss between the qubits to be entangled, using laser pulses routed in-plane through waveguides that thread among the cavities. Individual qubits can be coupled to the waveguides via optically-controlled tuning of resonant frequencies, accomplished using MEMS micromirror arrays and out-of-plane lasers.

Figure 1 shows our basic layout. The smallest circular cavities include the quantum dots used for qubits, and are assigned three low-level roles in the system: lattice qubits (wedged between larger circular waveguides), ancillae for teleported gate circuits (between large circles and racetrack-shaped waveguides), and transceivers (between racetrack-shaped waveguides and linear waveguides). The large, circular waveguides are used for high-fidelity gates between ancillae and lattice

qubits, requiring no additional purification.

The linear waveguides are arranged in alternating columns for logic gates and long-distance communication via teleportation and purification. The teleportation waveguides, flanked by many transceiver qubits, initially create low-fidelity Bell pairs over large vertical distances, between neighboring columns, or between chips, depending on the setting of optical switches (not shown in the figure; see the complete architecture in Ref. [5]). Assuming 0.1dB optical loss for in-column connections and 0.4dB between neighboring columns, our calculations show that the respective connections require approximately 100 and 1,000 pulses in the teleporation waveguides, including purification.

The master architecture is a quantum multicomputer, consisting of thousands of chips arranged in an $H \times 1$ array with each chip coupled to left and right neighbors through switched extensions of the teleportation waveguides. Previous studies have shown that this wide, shallow multicomputer arrangement works well for systems using CSS codes [6]; in this architecture, sections of lattice are stitched together in a distributed fashion across device boundaries, likewise avoiding the need for topologically complex inter-node switching fabrics [3].

3 Discussion

The architecture presented here has numerous strengths, including the explicit planning for heterogeneous communication and low yield. The lattice maintenance process is resource-efficient and highly flexible, minimizing the number of ancillae required for plaquette syndrome measurement. The rich interconnect supports software-defined lattice arrangements, allowing defective qubits to be mapped out of the system easily.

Our architecture has two principal weaknesses. First, despite the high-speed repetition rate of signals (specified as 10GHz) in the logic and teleportation waveguides, the large number of pulses required in each vertical column still results in a complete lattice refresh time of $\sim$50μsec. This in turn demands $\sim$50msec memory coherence time and results in Toffoli gate times that are also $\sim$50msec, slow but tolerable for our purposes. Second, although the basic structures of cavities and waveguides are large by modern VLSI standards, the spacing between cavities and waveguides and the surface roughness of optical components require very exacting fabrication. Adequate fabrication is currently achievable only with e-beam lithography, which is fast enough for small test devices but too slow (hence, expensive) for the production levels required for large-scale systems.

Our investigations produced a clear imperative that was unexpected, but logical in hindsight: pursue quality of components at the expense of yield. Within our system, a yield of only a few percent of qubits declared "functional" will allow lattice-building experiments, and a yield of 40% will allow the construction of supercomputer-scale systems capable of using Shor's algorithm to factor kilobit or larger numbers in a timescale of months, many times faster than classical systems. We believe that this minimum yield is the lowest yet reported.

Finally, this work has helped to establish targets for various experimental values: the 50msec memory lifetime mentioned above, a teleported gate error rate of 0.2%, and the demanding nature of optical communication. With our architecture and choice of gate mechanism, changes in switch loss of less than half a decibel result in an order of magnitude increase in the number of required pulses and corresponding negative effects on performance.

Our overall architecture bears out the intuitive idea that richer interconnectivity among components will improve the robustness of the system. By using a multi-level strategy incorporating purification and appropriate choices of physical gate sequences, heterogeneity and physical defects can be managed in an integrated fashion.

Acknowledgements

This research is supported in part by the Japan Society for the Promotion of Science (JSPS) through its "Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program)." This work was supported in part by National Science Foundation CCF0829694, with partial support by MEXT and NICT. We acknowledge the support of the Australian Research Council, the Australian Government, and the US National Security Agency (NSA) and the Army Research Office (ARO) under contract number W911NF-08-1-0527. The authors thank Shinichi Koseki for fabricating and photographing the test structure.

References

[1] J. Cirac, A. Ekert, S. Huelga, and C. Macchiavello. Distributed quantum computation over noisy channels. *Physical Review A*, 59:4249, 1999.

[2] J. Dehaene, M. Van den Nest, B. De Moor, and F. Verstraete. Local permutations of products of Bell states and entanglement distillation. *Physical Review A*, 67(2):22310, 2003.

[3] A. Fowler, A. Stephens, and P. Groszkowski. High threshold universal quantum computation on the surface code. *Physical Review A*, 80:052312, 2009.

[4] R. Raussendorf, J. Harrington, and K. Goyal. Topological fault-tolerance in cluster state quantum computation. *New Journal of Physics*, 9:199, 2007.

[5] R. Van Meter, T. D. Ladd, A. G. Fowler, and Y. Yamamoto. Distributed quantum computation architecture using semiconductor nanophotonics. *International Journal of Quantum Information*, 8:295–323, 2010. preprint available as arXiv:0906.2686v2 [quant-ph].

[6] R. Van Meter, W. J. Munro, K. Nemoto, and K. M. Itoh. Arithmetic on a distributed-memory quantum multicomputer. *ACM Journal of Emerging Technologies in Computing Systems*, 3(4):17, Jan. 2008.

†Present address: HRL Laboratories, LLC, 3011 Malibu Canyon Rd., Malibu, CA 90265