# Quantum Switching and Quantum Merge Sorting

Sheng-Tzong Cheng and Chun-Yen Wang

***Abstract*—This paper proposes a quantum switching architecture that can dynamically permute each input quantum data to its destination port to avoid using the fully connected networks. In addition, in order to reduce the execution time of the quantum switching, an efficient quantum merge sorting (QMS) algorithm that provides a parallel quantum computation is also developed. The quantum switching utilizes the QMS algorithm as a subroutine so that the total running time can be reduced to polylogarithmic time. Furthermore, to evaluate the feasibility of the quantum switching, we also define three different kinds of performance factors that can be used to estimate the complexity in implementation and the time delay in execution for quantum instruments. From the evaluation results, it can be seen that the proposed quantum switching is feasible in practice.**

***Index Terms*—Quantum circuits, quantum computation, quantum permutation, quantum sort, quantum switching.**

## I. Introduction

QUANTUM computation and *quantum information science* (QIS) [1], [2] that study the exploration of quantum mechanics provides us advanced tools to solve difficult problems in traditional Turing machines. The fundamental element in the quantum field can be any particle with two states that coexist together such as a photon, atom, electron, or subatomic particle [3]. All of them are under the nano-scale. Thus, QIS is often considered to be a basis for further research and innovations in the future development of nanotechnology.

In literature, many outstanding algorithms have been developed to illustrate that quantum-based computers are more powerful than traditional computers. For example, Shor [4] presents a quantum algorithm that factors a composite integer exponentially faster than the best classical algorithms do, and Grover [5] presents a quantum algorithm that searches an item from an unstructured list polynomial speedup over classical algorithms [6]. Due to its potential impact and significance, QIS has drawn more and more attention in the recent decade.

One of the most important applications in QIS is the quantum communication that is a mechanism for transferring quantum data from one location to another. Among them, *quantum wire* [7] is one of such mediums for transporting quantum information. At present, many remarkable research issues based upon the quantum-wire configuration such as quantum teleportation [8], quantum cryptography [9], [10], quantum repeater [11], and quantum networks [12] have been raised. These results have driven the QIS field further into real-world applications.

Manuscript received December 9, 2004; revised May 25, 2005. This paper was recommended by Associate Editor P. Nilsson.

The authors are with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan, R.O.C. (e-mail: stcheng@mail.ncku.edu.tw; eric@csie.ncku.edu.tw).

Digital Object Identifier 10.1109/TCSI.2005.856669

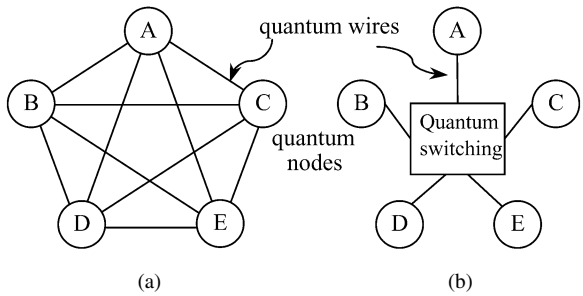


Fig. 1. Quantum network architecture. (a) Fully connected. (b) Quantum switching.

Many views of the future of QIS indicate that the need for basic quantum-wire components would become more essential due to the explosive growth of quantum devices. As shown in Fig. 1(a), in order to ensure that any two quantum devices can communicate quantum data with each other, it is necessary to employ a quantum wire between them. However, this results in a fully connected quantum network that would require $O(n^2)$ quantum wires to construct, where $n$ is the total number of quantum nodes. In this paper, a novel quantum switching architecture is proposed to avoid the fully connected quantum networks. As shown in Fig. 1(b), with the aid of quantum switching, quantum nodes only need to deliver their quantum data to the quantum switching, and the quantum switching would switch it to its destination port. Thus, each quantum node only requires one quantum wire to connect with quantum switching, so the number of quantum wires can be reduced to $O(n)$.

The proposed quantum switching allows each input quantum data to be permuted to its corresponding destination port in parallel dynamically. Moreover, to minimize the time delay in executing the quantum switching, an efficient quantum merge sorting (QMS) algorithm is developed in this paper. Although it has been shown that quantum computers, in general, cannot greatly outperform classical computers for the task of sorting [13], this paper makes use of the parallel computation technique [14] to reduce the running time of the QMS algorithm so that it only takes time complexity $O(\log^2 n)$ to sort $n$ quantum elements.

Because the proposed quantum switching utilizes the efficient QMS algorithm, the total execution time can be reduced to polylogarithmic time. In addition, in order to assess the feasibility of the quantum switching, we also define three different performance indexes to evaluate the effectiveness of quantum instruments. According to the results, it can be seen that the proposed quantum switching is feasible to put into practice. This paper represents one of the few attempts to solve the dynamic quantum switching in the literature.

The remainder of this paper is organized as follows. In Section II, the general concepts about quantum computation are in-

troduced. Related research is surveyed as well. In Section III, the QMS algorithm that provides a parallel quantum computation is presented. In Section IV, the quantum switching architecture is proposed to switch quantum data to its corresponding port. Finally, conclusions are drawn in Section V.

## II. Background and Related Work

In this section, we introduce definitions and terminologies used in describing the proposed mechanism throughout this paper. In-depth treatments of the notations can be found in the literature [1], [15].

### A. Quantum States

Analogous to classical bits, the underlying unit in QIS is a *quantum bit* (*qubit*). However, unlike a classical bit that must be in a state either 0 or 1, a qubit can be in both state $|0\rangle$ and state $|1\rangle$ at the same time. This special phenomenon in quantum computing is called *superposition*. In general, a qubit state is often represented as a linear combination of these two states

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

where $\alpha$ and $\beta$ are complex numbers and $||\alpha||^2 + ||\beta||^2 = 1$.

Note that it is impossible for us to determine whether a qubit is in state $|0\rangle$ or $|1\rangle$ by examining the values of $\alpha$ and $\beta$. However, when we *measure* a qubit in a superposition state, the entire qubit system would collapse into one of its bases (e.g., $|0\rangle$ or $|1\rangle$). As for which state we would obtain, it is determined by the absolute square of its coefficient. That is, we get the qubit in state $|0\rangle$ with probability $||\alpha||^2$ or in state $|1\rangle$ with probability $||\beta||^2$. Thus, $\alpha$ and $\beta$ are called *probability amplitudes*.

Furthermore, a multiple-qubit system can be built up by composing multiple independent qubits. To compose two or more qubit systems together, the *tensor product* operator $\otimes$ is adopted. For example, consider a two-qubit system composed of two single-qubit systems $|\phi\rangle = a|0\rangle + b|1\rangle$ and $|\psi\rangle = c|0\rangle + d|1\rangle$. Then, the entire two-qubit system is usually represented as

$$|\phi\rangle \otimes |\psi\rangle = |\phi\psi\rangle = ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle. \tag{2}$$

### B. Quantum Unitary Gates

In digital information processing, a basic circuit combined with some fundamental logic operators so as to complete a specific task is called a logic gate, such as the classical NOT gate, AND gate, OR gate, XOR gate, and so on. Analogously, in the quantum realm, a set of operators that can manipulate a quantum system to accomplish a specific computation is called a *quantum gate*. In this section, we introduce four quantum gates that are in common use throughout this paper.

Similar to the classical NOT gate, the quantum NOT gate applied on a single qubit can be used to flip its states $|0\rangle$ and $|1\rangle$. As shown in Fig. 2(a), when we send a qubit $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$ through the quantum NOT gate, then the corresponding output would be $|\phi_{\text{output}}\rangle = \beta|0\rangle + \alpha|1\rangle$. Besides, the quantum NOT gate is usually called as a *Pauli-X* gate. Fig. 2(b) shows the circuits of the *Controlled-NOT* (CNOT) gate for processing two

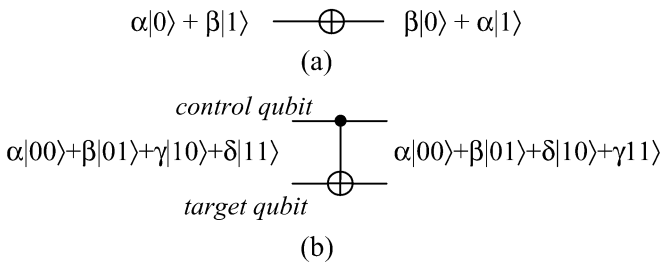


Fig. 2. (a) Quantum NOT gate and (b) CNOT gate.

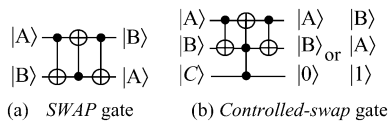


Fig. 3. Quantum SWAP gate and controlled-swap gate.

qubits. It can be seen from Fig. 2(b) that the quantum CNOT gate has two inputs. One is the control qubit and the other is the target qubit. When the control qubit is set to $|0\rangle$, the CNOT gate does nothing. On the other hand, if the control qubit is set to $|1\rangle$, then it would apply the NOT gate to the target qubit. In other words, the CNOT gate can be used to exchange $|10\rangle$ and $|11\rangle$.

Besides, one more important quantum operator is the quantum SWAP gate. The SWAP gate applied on a two-qubit system can be used to do the transposition. More precisely, it changes a two-qubit system from $|A\rangle \otimes |B\rangle$ to $|B\rangle \otimes |A\rangle$. As drawn in Fig. 3(a), the SWAP gate can be implemented by three layers of CNOT gates. However, because of its importance to quantum communication, SWAP gate is often seen as a basic quantum gate.

The last quantum operator we introduce in this subsection is the controlled-swap gate (or the Fredkin gate). As depicted in Fig. 3(b), the controlled-swap gate has three input qubits, $|A\rangle$, $|B\rangle$, and $|C\rangle$. The qubit $|C\rangle$ is called a control qubit because it controls what happens to the other two qubits. When the control qubit $|C\rangle$ is set to $|0\rangle$, it does nothing. On the other hand, if the control qubit $|C\rangle$ is set to $|1\rangle$, then the SWAP gate is applied to the target qubits (i.e., $|A\rangle$ and $|B\rangle$ are swapped).

In quantum circuits, time evolution goes from left to right. Furthermore, each quantum operation is defined to be certainly revertible, so it can be done in principle without consuming any energy [16]. This particular reversibility property is also put to use by the proposed quantum switching so as to clean up some garbage bits. More details are described clearly in Section IV-B.

### C. Quantum Digital Switch

In literature, Tsai and Kuo [17] present a *digital switching* that can switch digital data in the quantum domain. The basis idea of their digital switching is to apply the property that any general cycle can be performed by using six layers of CNOT gates [5]. This results in their digital switching being able to be implemented in a constant-time complexity [18].

However, the *control module* in their digital switching was absent. The control module is responsible for controlling the switching configuration so that all of the input data in each individual time can be switched to their destination ports correctly. In order to dynamically control the switching module in each

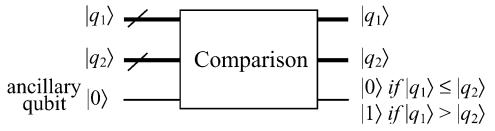


Fig. 4. Comparison gate.

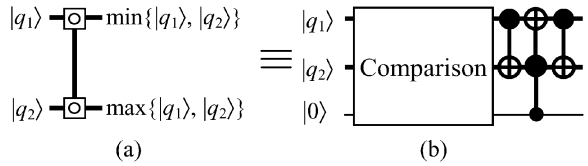


Fig. 5. Quantum circuits for a quantum *comparator* gate.

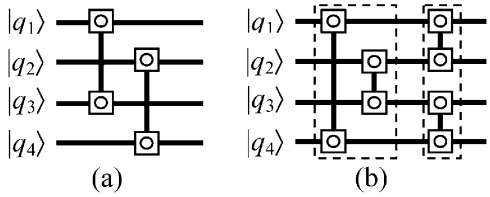


Fig. 6. Examples for quantum comparator gates.

individual time, we propose a new quantum switching architecture with the control module in this paper.

In addition, Shukla *et al.* [19] present a *quantum Banyan network* that only requires logarithmic execution time to switch input data. However, the quantum Banyan network is not revertible since it needs to discard some dummy packets. In addition, the most serious problem in the quantum Banyan network is that some input quantum information would be lost. Further, in the worst case, it is possible for the quantum Banyan network to lose $n-2$ input quantum packets, where $n$ is the total number of input packets. According to the no-cloning theorem [20], [21], these lost quantum data cannot be recovered. This would result in an unreliable quantum network.

The proposed quantum switching can overcome these drawbacks. The quantum switching is designed to be revertible. Moreover, no input data would be lost in executing quantum switching. The quantum operations for implementing the quantum switching are described clearly in Section IV. Since the quantum switching is constructed by applying an efficient quantum sorting algorithm, we briefly discuss some of the existing issues in quantum sorting in the following.

### D. Quantum Sorting Algorithm

In classical information theory, the traditional sorting problem can be formulated as follows. Given a sequence of $n$ numbers $Q = \langle q_1, q_2, \ldots, q_n\rangle$, output a permutation $Q' = \langle q'_1, q'_2, \ldots, q'_n\rangle$ of the input sequence such that the list $Q'$ is in nondecreasing order [13], [22]. However, unlike the classical bits, there exists a special superposition phenomenon in the quantum field. Thus, it is necessary to apply the so-called *black box* model to determine the *large* or *small* relationship between two input quantum numbers.

Fig. 4 shows the sketch of the *comparison* gate that is commonly used to define the order relation $\le$ between two quantum numbers [1]. As shown in Fig. 4, the comparison gate has three inputs: two quantum numbers $|q_1\rangle$, $|q_2\rangle$ and an ancillary qubit. If $|q_1\rangle$ is *greater* than $|q_2\rangle$, then the comparison gate flips the state of the ancillary qubit, else it does nothing. For clarity, in this paper, we also adopt the *thick line* as in Fig. 4 to represent that the quantum number is composed of multiple qubits and use the *thin line* to stand for a single qubit.

In literature, Farhi *et al.* [23] propose a quantum algorithm that can search an element in a sorted list of $n$ elements as a subroutine in *Insertion sort* by querying the comparison gates roughly $0.526\log_2 n$ times. This achieves that a quantum computer needs $0.526n\log_2 n + O(n)$ queries to sort $n$ elements. However, is it possible to improve the performance of the quantum sorting algorithm by decreasing the upper bound for the ordered search? Unfortunately, this idea cannot be put into practice because the lower bound $(\log_2 n/12) - O(1)$ for the ordered search problem has been verified by Ambainis [24].

After that, Høyer *et al.* [13] showed that any comparison-based quantum sorting algorithm needs $\Omega(n\log n)$ queries to the comparison gates, which is already achieved by classical sorting algorithms. In other words, it shows that quantum algorithms can only achieve a constant speedup over classical algorithms for the sorting problem. Besides, Arul [25] also demonstrates that nonorthogonal quantum states cannot be compared or sorted. Also, Klauck *et al.* [26], [27] study the lower bound on the time–space tradeoffs for sorting $n$ elements.

In this paper, we present a QMS algorithm that provides a parallel computation for our quantum switching. The QMS algorithm is constructed by the *comparator* gates (or the *control-and-swap* operation in some publications [1]) defined in Fig. 5(a) completely. As drawn in Fig. 5(b), the quantum comparator gate is composed of a comparison gate and controlled-swap gates. The former comparison gate compares the input elements $|q_1\rangle$ with $|q_2\rangle$ and changes the ancilla state if $|q_1\rangle$ is greater than $|q_2\rangle$. The latter controlled-swap gates are responsible for permuting $|q_1\rangle$ and $|q_2\rangle$ if the ancilla state is set to $|1\rangle$. In other words, the whole comparator gate is the quantum operation that can be used to sort two quantum numbers.

For simplicity, in this paper, we use the abbreviated notion $C[i,j]$ to indicate the comparator gate on the qubits $|q_i\rangle$ and $|q_j\rangle$. Thus, the quantum circuits shown in Fig. 6(a) can be formulated as $C[1,3]\otimes C[2,4]$, and the quantum circuits depicted in Fig. 6(b) can be represented by $(C[1,2]\otimes C[3,4])\circ(C[1,4]\otimes C[2,3])$.

## III. Quantum Merge Sorting

In this section, the QMS$(n)$ algorithm for sorting $n$ quantum numbers is presented. In the beginning, we consider that the number of inputs $n$ is a power of two (i.e., $n = 2^k$). As for a general $n$, we discuss it in Section III-D.

Analogous to the classical merge sorting algorithm [22], the QMS algorithm also follows a divide-and-conquer approach and is a recurrence in structure. In fact, the QMS$(n)$ algorithm can be defined in the following form:

$$\text{QMS}(n) = \begin{cases} \text{Merge}(n)\circ\left(\text{QMS}\left(\frac{n}{2}\right)\otimes\text{QMS}\left(\frac{n}{2}\right)\right), & \text{if } n > 1 \\ I, & \text{if } n = 1 \end{cases} \tag{3}$$

where $I$ denotes the identity operation.

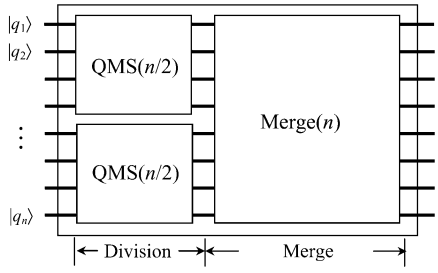


Fig. 7. Skeleton diagram of the QMS$(n)$ architecture.

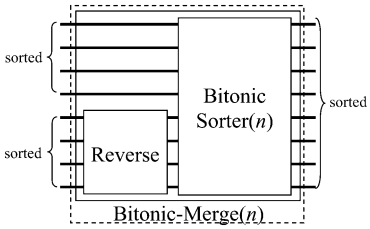


Fig. 8. Skeleton diagram for quantum bitonic merge.

We can depict the quantum circuits for the QMS$(n)$ as shown in Fig. 7. It can be seen from Fig. 7 that the QMS algorithm is composed of two phases: the *division* phase and the *merge* phase. During the division phase, the QMS$(n)$ algorithm is recursively performed to transform $n$ unsorted quantum numbers into two sorted lists. Afterwards, these two sorted lists are combined into one increasing sequence during the merge phase. In addition, this paper develops a special merge strategy, *quantum bitonic merge*, to reduce the running time of the QMS algorithm. The details for constructing the quantum bitonic merge are described in the following.

### A. Quantum Bitonic Merge

In this subsection, we start out to present the quantum bitonic merge to combine two sorted quantum lists into an ordered sequence. The skeleton for the quantum bitonic merge is depicted in Fig. 8. As shown, the quantum bitonic merge can be further split into two components: *reverse* operation and *bitonic sorter*.

The reverse operation is responsible for transforming two sorted quantum sequences into one *quantum bitonic sequence* while the bitonic sorter can sort any bitonic sequence [28]. A quantum bitonic sequence is defined as a quantum sequence that is either monotonically increasing and then monotonically decreasing or monotonically decreasing and then monotonically increasing. For example, suppose that the ordered relation defined in the comparison gate satisfies $|\psi_1\rangle \le |\psi_2\rangle \le |\psi_3\rangle \le |\psi_4\rangle \le |\psi_5\rangle \le |\psi_6\rangle \le |\psi_7\rangle \le |\psi_8\rangle$. Then, the quantum sequences $\{|\psi_1\rangle, |\psi_3\rangle, |\psi_6\rangle, |\psi_8\rangle, |\psi_2\rangle\}$ and $\{|\psi_8\rangle, |\psi_5\rangle, |\psi_2\rangle, |\psi_4\rangle, |\psi_6\rangle, |\psi_7\rangle\}$ are both bitonic sequences. Besides, a monotonically increasing or decreasing sequence is also a bitonic sequence.

The quantum circuit of the quantum bitonic sorter is designed for sorting any quantum bitonic sequence. Fig. 9 demonstrates the quantum circuits for quantum bitonic sorter with $n$ input elements. It can be seen from Fig. 9 that the quantum bitonic sorter is constructed by applying $n/2$ comparator gates in parallel and then recursively performing the Quantum Bitonic-Sorter$(n/2)$.

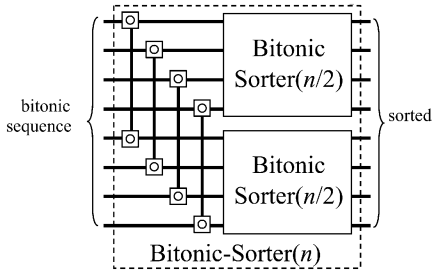


Fig. 9. Quantum circuits for quantum Bitonic-Sorter$(n)$.

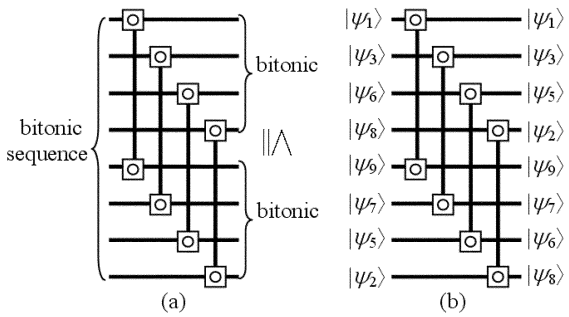


Fig. 10. First stage of the quantum bitonic sorter.

Thus, we can express the quantum bitonic sorter in the following mathematical form:

$$\text{Sorter}(n) = \begin{cases} \left(\text{Sorter}\left(\frac{n}{2}\right) \otimes \text{Sorter}\left(\frac{n}{2}\right)\right) \circ \overset{n/2}{\underset{i=1}{\otimes}} C\left[i, \frac{n}{2}+i\right], & \text{if } n>1 \\ I, & \text{if } n=1. \end{cases} \tag{4}$$

The basic idea of the quantum bionic sorter for sorting any bitonic sequence is to separate input bitonic sequences into two bitonic sequences. Also, by recursively performing the quantum bitonic sorter in parallel, we can sort the original bitonic sequence ultimately. More precisely, as shown in Fig. 9, the former part of the Quantum Bitonic-Sorter$(n)$ is made up of $n/2$ comparator gates. Those comparator gates compare each input $i$ with input $n/2+i$ for $i = 1, \ldots, n/2$ in parallel. The purpose of those comparator gates is to partition the original bitonic sequence into two halves such that: 1) both halves are bitonic and 2) any quantum number in the first half is smaller than or equal to all quantum numbers in the second half [see Fig. 10(a)].

A simple example is shown in Fig. 10(b). Consider a quantum sequence $\{|\psi_1\rangle, |\psi_3\rangle, |\psi_6\rangle, |\psi_8\rangle, |\psi_9\rangle, |\psi_7\rangle, |\psi_5\rangle, |\psi_2\rangle\}$ with eight quantum numbers. Suppose that the ordered relation defined in the comparison gate satisfies that $|\psi_1\rangle \le |\psi_2\rangle \le |\psi_3\rangle \le |\psi_4\rangle \le |\psi_5\rangle \le |\psi_6\rangle \le |\psi_7\rangle \le |\psi_8\rangle$. By the above definition, we know that this input quantum sequence is a bitonic sequence. Then, after performing the quantum circuits in Fig. 10(a), we can get two sequences $\{|\psi_1\rangle, |\psi_3\rangle, |\psi_5\rangle, |\psi_2\rangle\}$ and $\{|\psi_9\rangle, |\psi_7\rangle, |\psi_6\rangle, |\psi_8\rangle\}$. It is clear that these two output sequences are both bitonic and each quantum number in the first half is *smaller* than that in the second half. Thus, by following the same procedure continuously, we can ultimately get a sorted sequence. As a result, we can sort any bitonic sequence by performing the quantum circuits in Fig. 9.

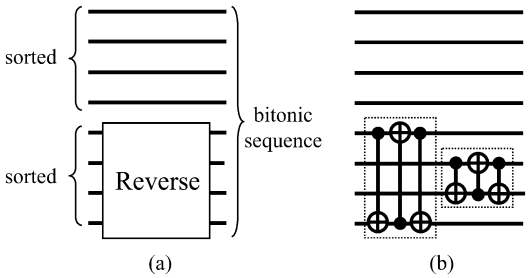


Fig. 11. Quantum circuits for quantum reverse operation.

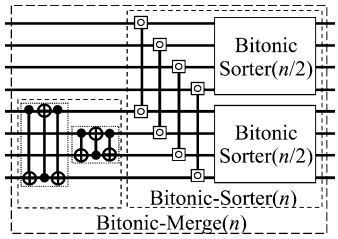


Fig. 12. Skeleton diagram for quantum Bitonic-Merge$(n)$.

The quantum bitonic sorter only accepts bitonic sequences. However, it can be seen from Fig. 7 that the output sequences from the division phase are not bitonic, but are two sorted sequences. Therefore, it is necessary to employ the reverse operation to convert the ordering of one sorted sequence. Consequently, the output turns out to be a bitonic sequence.

As shown in Fig. 11(a), two sorted nondecreasing quantum sequences can be easily transformed into a bitonic sequence by reversing the order of one of these two sequences. More precisely, suppose that $\{|q_1\rangle, |q_2\rangle, \ldots, |q_m\rangle\}$ and $\{|r_1\rangle, |r_2\rangle, \ldots, |r_n\rangle\}$ are two nondecreasing quantum sequences. If we reverse the order of the latter sequence, then it is clear that the new quantum sequence $\{|q_1\rangle, |q_2\rangle, \ldots, |q_m\rangle, |r_n\rangle, |r_{n-1}\rangle, \ldots, |r_1\rangle\}$ is bitonic.

The reverse operation that reverses the order of any input list can be easily implemented by employing $n/4$ swap gates as shown in Fig. 11(b). Each input $n/2+i$ is swapped with input $n-i+1$ for $i = 1, 2, \ldots, n/2$. Note that those $n/4$ swap gates can be executed in parallel by using only three layers of CNOT gates.

Now we can substitute the quantum reverse operation (Fig. 11) and bitonic sorter (Fig. 9) into quantum bitonic merge (Fig. 8) to obtain the quantum bitonic merge circuits, as depicted in Fig. 12. Inputs of two sorted sequences from the division phase are converted into a bitonic sequence by the reverse operation. Also, the quantum bitonic sorter can sort any bitonic sequence. Thus, two sorted quantum sequences can be easily merged into an order list through the quantum bitonic merge in Fig. 12.

Nevertheless, the quantum circuits in Fig. 12 can be further simplified. The left-hand quantum circuits of the quantum Bitonic-Merge$(n)$ are taken out as in Fig. 13(a). As mentioned above, the quantum circuits shown in Fig. 13(a) can transform two sorted quantum sequences into two bitonic sequences such that the output numbers in the top half are at least as small as that in the bottom half. In fact, the quantum circuits in Fig. 13(a) are equivalent to those in Fig. 13(b). Comparing Fig. 13(a) with Fig. 13(b), each input $i$ in Fig. 13(a) is compared with input $n/2+i$ that is swapped from input $n-i+1$, while each input $i$ in Fig. 13(b) is directly compared with input $n-i+1$, and then the reverse operation is applied to conform to the output of Fig. 13(a).

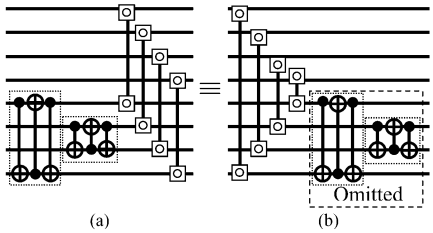


Fig. 13. Quantum circuit simplification.

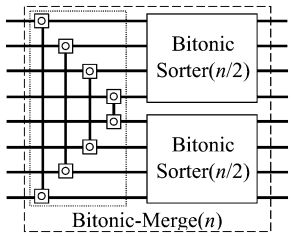


Fig. 14. Quantum circuits for quantum Bitonic-Merge$(n)$.

Since the quantum circuits in Fig. 13(a) and (b) are equivalent, the circuits in Fig. 13(b) can also convert two sorted quantum sequences into two bitonic sequences such that every quantum number in the top half is at least as small as each one in the bottom half. In addition, it can be observed that, even if we omit the reverse operation in Fig. 13(b), the outputs still satisfy this property. This is because the reversal of any bitonic sequence is also bitonic. Therefore, we can only perform $n/2$ comparator gates in parallel to substitute the $n/4$ swap gates and $n/2$ comparator gates in Fig. 12.

After replacing the quantum circuits in Fig. 13(a) with the quantum circuits in Fig. 13(b), we can get the new quantum Bitonic-Merge$(n)$ as depicted in Fig. 14. In quantum Bitonic-Merge$(n)$, each input $i$ is compared with input $n-i+1$ for $i = 1, 2, \ldots, n/2$ to produce two bitonic sequences and then the top half and bottom half can be sorted in parallel by the quantum bitonic sorter. As a result, in terms of the mathematic equations, the quantum Bitonic-Merge$(n)$ can be written as

$$\text{Merge}(n) = \begin{cases} \left(\text{Sorter}\left(\frac{n}{2}\right) \otimes \text{Sorter}\left(\frac{n}{2}\right)\right) \circ \bigotimes_{i=1}^{n/2} C[i, n-i+1], & \text{if } n > 1 \\ I, & \text{if } n = 1. \end{cases} \tag{5}$$

### B. QMS

In this subsection, we illustrate the QMS algorithm for sorting eight quantum elements. As mentioned above, the QMS algorithm is a recurrence in structure. Thus, after recursively expanding the QMS(8) algorithm, we can get the infrastructure constructed by the quantum bitonic merge as shown in Fig. 15.

In addition, as depicted in Fig. 14, for each quantum Bitonic-Merge$(n)$, it can be implemented by executing $n/2$ comparator gates and two quantum Bitonic-Sorter$(n/2)$'s in parallel. Thus, after replacing the quantum bitonic merge by the

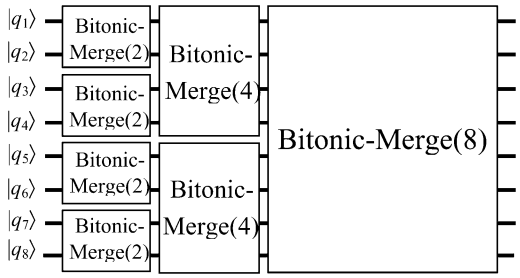


Fig. 15. Recursively expanding the QMS(8).

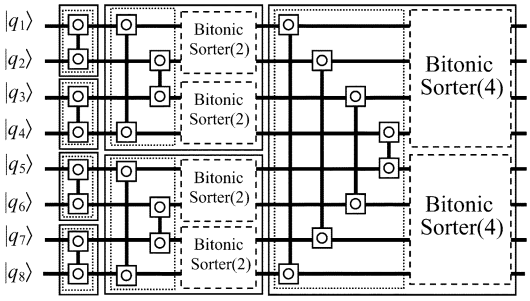


Fig. 16. Replacing the quantum bitonic-merge circuits.

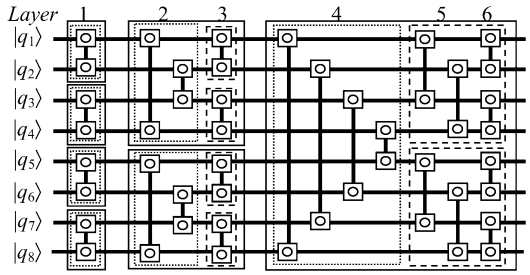


Fig. 17. All QMS(8) circuits.

circuits in Fig. 14, we can obtain the construction as depicted in Fig. 16.

Finally, as drawn in Fig. 9, the quantum Bitonic-Sorter$(n)$ is also a recursive construction, and it can be performed by applying $n/2$ comparator gates and two Bitonic-Sorter$(n/2)$ operations in parallel. Thus, we can substitute Fig. 9 into Fig. 16, and then all of the QMS(8) circuits can be carried out as shown in Fig. 17.

### C. Performance Evaluation

In this subsection, the performance of the QMS algorithm is studied. Let $NQ(n)$, $NM(n)$, and $NS(n)$ denote the number of comparator gates that are needed to implement the QMS$(n)$, Merge$(n)$, and Sorter $(n)$ circuits, respectively. Thus, according to (3)–(5), it is clear that

$$NQ(n) = 2NQ\left(\frac{n}{2}\right) + NM(n) \tag{6}$$

$$NM(n) = 2NS\left(\frac{n}{2}\right) + \frac{n}{2} \tag{7}$$

$$NS(n) = 2NS\left(\frac{n}{2}\right) + \frac{n}{2} \tag{8}$$

for $n > 1$, and $NQ(1) = NM(1) = NS(1) = 0$.

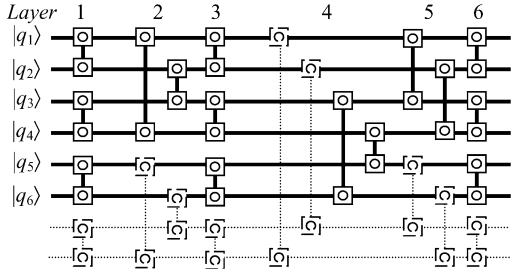


Fig. 18. Quantum circuits for QMS(6).

By applying the Master theorem [22] to the above equations, we know that

$$NQ(n) = \Theta(n \log n). \tag{9}$$

From (9), it can be seen that the QMS algorithm still requires $\Theta(n \log n)$ queries to comparison gates that matches the lower bound by Høyer [13]. However, the purpose of the QMS algorithm is not to outperform other quantum sorting algorithms, but to provide an efficient parallel quantum computation to decrease the whole running time of quantum sorting algorithms.

Let $TQ(n)$, $TM(n)$, and $TS(n)$ denote the time complexity of the QMS$(n)$, Merge$(n)$, and Sorter $(n)$ circuits, respectively. Similarly, based on (3)–(5), it can be known that

$$TQ(n) = TQ\left(\frac{n}{2}\right) + TM(n) \tag{10}$$

$$TM(n) = TS\left(\frac{n}{2}\right) + TC(k) \tag{11}$$

$$TS(n) = TS\left(\frac{n}{2}\right) + TC(k) \tag{12}$$

for $n > 1$, where $TC(k)$ is the time complexity of the comparator gate with two $k$-bit input numbers, and $TQ(1) = TM(1) = TS(1) = 0$.

After expanding the recursive equations, we obtain

$$TQ(n) = \Theta((\log n)^2 TC(k)). \tag{13}$$

Consequently, the QMS$(n)$ algorithm can sort $n$ quantum numbers in parallel and only take time $\Theta((\log n)^2)$ layers of comparator gates. In other words, the QMS algorithm achieves the polylogarithmic time complexity. Thus, it can be employed in constructing some quantum instruments with critical time delay such as quantum switching. The details of the quantum operations required to implement the quantum switching are described in Section IV.

### D. General Cases

In this subsection, we consider the QMS$(n)$ algorithm for the case in which $n$ is not exactly a power of two. The first step to construct such a QMS$(n)$ operation is to implement the QMS$(2^r)$ circuits that can be done by following the methodology described in Section III-B, where $r = \lceil \log_2 n \rceil$. Afterwards, we omit all of the quantum gates performed on the last $2^r - n$ quantum elements. Then, the remaining quantum circuits are the QMS$(n)$ circuits that can sort $n$ quantum numbers.

An example for QMS(6) circuits is illustrated in Fig. 18. To construct the QMS(6) circuits, the first step is to build the quantum circuits for QMS$(2^3)$ due to $\lceil \log_2 6 \rceil = 3$ Afterwards,

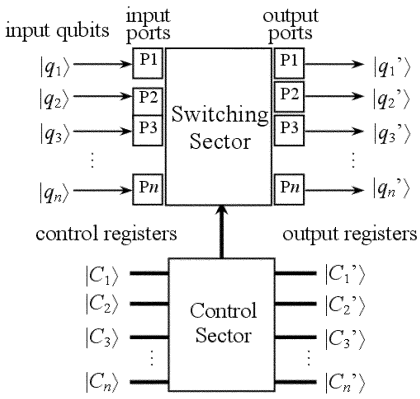


Fig. 19. Sketch of the quantum switching architecture.

we remove all of the quantum gates that are operated on for the last two $(2^3 - 6 = 2)$ quantum numbers (the dotted part in Fig. 18). Then, the remaining quantum circuits are the QMS(6) circuits as depicted in Fig. 18.

The basic idea of the QMS$(n)$ algorithm for any natural number $n$ is to imagine that there are $2^r - n$ appended *very large* quantum numbers to meet the total sorted number of a power of two. By applying the method described in Section III-B, the QMS$(2^r)$ can be implemented. Since those $2^r - n$ appended quantum numbers are large enough that all of the comparator gates performed on them do not work, we can remove those comparator gates performed on them. Then, the remaining circuits are the quantum operator that can be employed to sort $n$ quantum elements.

From the complexity point of view, it can be seen from (9) and (13) that for a general $n$

$$\begin{aligned} NQ(2^{r-1}) < NQ(n) \le NQ(2^r) \\ \Rightarrow NQ(n) = \Theta(n \log n) \end{aligned} \tag{14}$$

$$\begin{aligned} TQ(2^{r-1}) < TQ(n) \le TQ(2^r) \\ \Rightarrow TQ(n) = \Theta((\log^2 n) TC(k)) \end{aligned} \tag{15}$$

where $r = \lceil \log_2 n \rceil$. In other words, it also consists of $\Theta(n \log n)$ comparator gates in depths $\Theta(\log^2 n)$, just as the computational complexity we demonstrated above.

## IV. Quantum Switching

In this section, the proposed quantum switching to avoid the fully connected networks is presented. The quantum switching is designed to permute each input data to its corresponding destination port dynamically. In addition, since quantum qubits cannot be replicated [20], [21], the *multicast* service is not supported by the proposed quantum switching. In other words, each input data are allowed to have exactly one destination port. On the other hand, if there exist two or more input qubits with the same destination port, it would result in the so-called *output collision*. Therefore, to prevent the quantum switching from output collision, we only consider the case that each input data and output port have one-to-one mapping.

Fig. 19 shows the sketch of the proposed quantum switching architecture. As shown, the quantum switching consists of two sectors: the *switching sector* and the *control sector*. The switching sector is responsible for permuting all of the input qubits into their corresponding output ports, and the control sector is responsible for controlling the switching sector so that the switching sector can accomplish its task correctly. Actually, the data permutation in the switching sector is dominated by the *control registers*. More precisely, consider the case that there is a qubit $|q_i\rangle$ input to the I/O port $P_i$ with the destination port $P_j$. To fulfill this switching task, we set the corresponding control registers $|C_i\rangle$ to be $|j\rangle$. After we set all of the control registers, by performing the quantum switching, all of the input data would be switched to their corresponding destination port correctly.

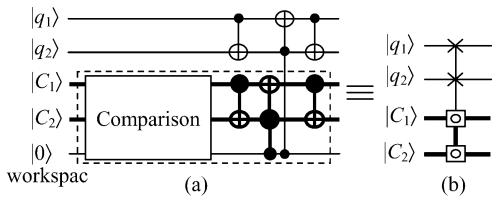


Fig. 20. Sketch of the partial comparator gate for quantum switching.

In essence, quantum switching can be viewed as a quantum sorting algorithm that sorts all of the input qubits $|q_i\rangle$ into an increasing order by their corresponding destination port number $|C_i\rangle$. As a result, the quantum switching architecture can be constructed easily by using the QMS algorithm presented in Section III. The details are described as follows.

### A. Quantum Switching Circuits

Before we present how quantum switching can be constructed by using the QMS algorithm, we define a particular *partial comparator gate* for the quantum switching first. As shown in Fig. 20(a), the partial comparator gate only compares the register $|C_1\rangle$ with the register $|C_2\rangle$. If the register $|C_1\rangle$ is smaller than or equal to the register $|C_2\rangle$, then it does nothing. Otherwise, the partial comparator gate flips the workspace qubit state (i.e., changes the state from $|0\rangle$ to $|1\rangle$) and swaps the registers $|C_1\rangle$ and $|C_2\rangle$ and the qubits $|q_1\rangle$ and $|q_2\rangle$. Actually, the partial comparator gate can be seen as a comparator gate appended with a controlled-swap gate. Therefore, for the purpose of simplicity, we adopt the abbreviated drawing in Fig. 20(b) to represent the partial comparator gate defined in Fig. 20(a).

Quantum switching can be seen as a kind of quantum sorting algorithm that sorts each input qubit by its destination port number. The first step of constructing the quantum switching is to establish the QMS$(n)$ circuits in the control sector. Then, for each comparator gate in the control sector, we append a controlled-swap gate on the corresponding input qubits.

An example for implementing the quantum switching with four I/O ports is illustrated in Fig. 21. As mentioned above, the quantum switching can be done by first constructing the QMS(4) circuits in the control sector as in Fig. 21(a). Afterwards, a controlled-swap gate is appended to each comparator gate on the corresponding input qubits. As drawn in Fig. 21(b), for the comparator gate applying on the registers $|C_i\rangle$ and $|C_j\rangle$, a controlled-swap gate is performing on the input qubits $|q_i\rangle$ and $|q_j\rangle$. After that, the quantum circuit is obtained as drawn in Fig. 21(b). Furthermore, it can be seen that, after performing the quantum circuits in Fig. 21(b), each input qubit is permuted to its corresponding destination port.

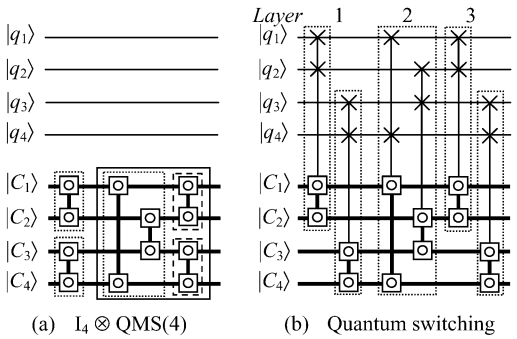


Fig. 21. Example for quantum switching circuits with four I/O ports. (a) $I_4 \otimes$ QMS$(4)$. (b) Quantum switching.

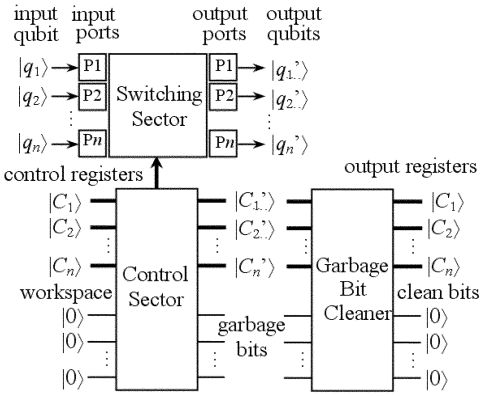


Fig. 22. Entire quantum switching architecture.

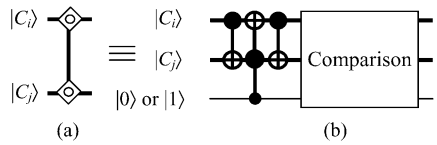


Fig. 23. Inverse of the comparator gate.

## B. Garbage-Bit Cleaner

In addition to the switching sector and control sector, it is necessary for the quantum switching to implement a *garbage-bit cleaner*. As depicted in Fig. 20(a), whenever we employ the partial comparator gate, we require an ancillary workspace to make the partial comparator gate revertible. However, those qubits would become garbage bits after the quantum switching. Therefore, to utilize those workspace qubits more efficiently, we implement a garbage-bit cleaner to clean those garbage bits so that they can be reused in the next round (Fig. 22).

As shown in Fig. 22, the garbage-bit cleaner can be roughly seen as the inverse of the control sector. Since the control sector is primarily constructed by the QMS algorithm, the garbage-bit cleaner can be done by implementing the inverse of the QMS algorithm. We employ the following methodology to build the inverse of the QMS operation. The first step is to implement the QMS circuits in reverse order, and the second step is to replace each comparator gate by the inverse of the comparator gate.

An example for implementing the inverse of the QMS(4) is illustrated here. We define the quantum logical gate drawn in Fig. 23(a) to be the inverse of the comparator gate. It is clear

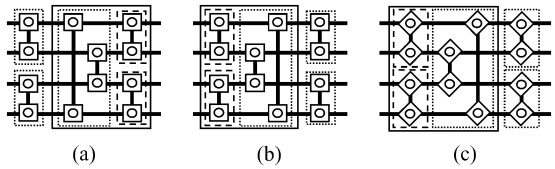


Fig. 24. Process to implement the garbage-bit cleaner.

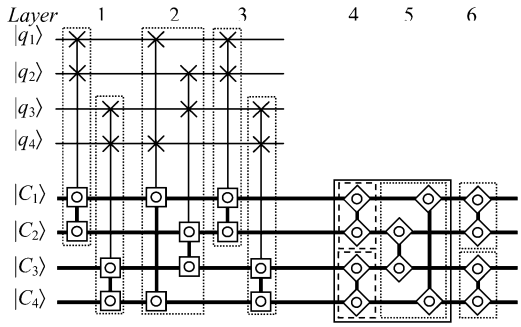


Fig. 25. Entire quantum switching circuits.

that, as shown in Fig. 23(b), the inverse of the comparator gate can be constructed by applying the comparator circuits in Fig. 5 in reverse order. Now, we can start out to build the garbage-bit cleaner. Fig. 24(a) shows the QMS(4) quantum circuits. To construct the inverse of the QMS(4) operation, we first implement the QMS(4) circuits in reverse order, as depicted in Fig. 24(b). After we replace each comparator gate in Fig. 24(b) by the inverse of the comparator gate, we get the inverse of the QMS(4), as depicted in Fig. 24(c). By substituting the quantum circuits in Figs. 21(b) and 24(c) into the quantum switching architecture in Fig. 22, we can get the whole quantum switching circuits as depicted in Fig. 25.

## C. Performance Evaluation

In this subsection, the performance of the quantum switching is studied. We define three factors to evaluate the effectiveness of the quantum switching: *space complexity*, *implementation complexity*, and *delayed time complexity*. Each of these plays an important role in evaluating the practicability of a quantum instrument. Space complexity and implementation complexity refer to the complexity in accomplishing the quantum instrument. Delayed time complexity represents the running time in execution. In practice, the delayed time complexity is more critical than the space complexity and implementation complexity since the proposed quantum switching is designed to be reused. The detailed definitions of those performance indexes are described as follows.

*Definition: The space complexity $SC(n)$ is defined as the total number of input qubits for the quantum switching with n I/O ports.* ■

As depicted in Fig. 22, the space complexity $SC(n)$ for the quantum switching contains $n$ input qubits, $n$ control registers, and extra workspace qubits. Since each control register is responsible for storing the destination port number of the corresponding input qubit, it requires at least $\log_2 n$ qubits. As for the number of workspace qubits, because we have to employ an additional workspace qubit for each partial comparator gate

and it can be seen from (9) that the quantum switching requires performing $\Theta(n \log n)$ comparator gates, we need $\Theta(n \log n)$ workspace qubits in total. Consequently

$$SC(n) = n + n \log_2 n + \Theta(n \log n) = \Theta(n \log n). \quad (16)$$

On the other hand, it is clear that quantum switching with $n$ I/O ports has to be able to handle $n!$ possible cases. Due to $n! = \Theta(2^{n \log n})$ [22], it is unavoidable for the quantum switching to apply $\Theta(n \log n)$ control bits to specify each possible switching case. This matches the space complexity of the proposed quantum switching. Thus, the proposed quantum switching is verified to be efficient in space consumption.

*Definition: The implementation complexity $IC(n)$ is defined as the number of quantum operations to implement the quantum switching with $n$ I/O ports.* ■

According to Fig. 25 and (9), it can be seen that the quantum switching with $n$ I/O ports is composed of $\Theta(n \log n)$ partial comparator gates and $\Theta(n \log n)$ inverses of comparator gates. In addition, as shown in Fig. 20(a), each partial comparator gate contains a comparison gate and $\log_2 n + 1$ controlled-swap gates. Also, the inverse of comparator gate can be constructed by $\log_2 n$ controlled-swap gates and a comparison gate, as depicted in Fig. 23(b). Furthermore, for the case of the proposed quantum switching, the comparison gate can be constructed by $\Theta(n \log n)$ quantum operations [29]. As a result, it is clear that

$$\begin{aligned} IC(n) &= \Theta\big[(n \log n)(\log n + (\log n + 1)) \\ &\quad + (n \log n)(\log n + \log n)\big] \\ &= \Theta(n \log^2 n). \end{aligned} \quad (17)$$

*Definition: The delayed time complexity $DT(n)$ is defined as the running time complexity of the quantum switching with $n$ I/O ports.* ■

It can be seen from Fig. 25 and (13) that the whole quantum switching can be implemented in $\Theta(n \log n)$ layers of partial comparator gates with a time complexity of $\Theta(n \log n)$ and $\Theta(n \log n)$ layers of inverses of comparator gates with a time complexity of $\Theta(n \log n)$. Therefore, we know that

$$DT(n) = \Theta[(\log^2 n)(\log n) + (\log^2 n)(\log n)] = \Theta(\log^3 n). \quad (18)$$

In other words, the quantum switching only takes $\Theta(n \log n)$ space consumption and can be implemented with $\Theta(n \log^2 n)$ quantum gates. Furthermore, in terms of time complexity, the time delay for executing the quantum switching only requires $\Theta(\log^3 n)$ operation time. This achieves a quantum switching that can be performed only in polylogarithmic time. Therefore, it turns out that the proposed quantum switching architecture is very scalable.

## V. Conclusion

Because many breakthroughs in quantum communication have been discovered, this results in the fact that the demand for establishing the communication networks becomes more and more urgent. To save the cost in building the quantum backbone networks, this paper proposes a quantum switching architecture that can be applied to switch each input quantum data to its corresponding destination port correctly. The proposed quantum switching employs an efficient QMS algorithm as a subroutine, so the total execution time can be reduced in polylogarithmic time. More precisely, the quantum switching only requires $\Theta(n \log n)$ space consumption and can be implemented with $\Theta(n \log^2 n)$ quantum gates. Furthermore, the quantum switching achieves a better performance in a time complexity of $\Theta(\log^3 n)$. Based on these advantages, it can be seen that the proposed quantum switching is feasible to be applied when constructing high-performance quantum networks.

## References


[1] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[2] D. DiVincenzo, "Quantum computation," *Science*, vol. 270, pp. 255–261, 1995.

[3] S. Lloyd, "A potentially realizable quantum computer," *Science*, vol. 261, pp. 1569–1571, 1993.

[4] P. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. IEEE Symp. Foundations Comput. Sci.*, 1994, pp. 124–134.

[5] L. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. 28th Annu. ACM Symp. Theory Computing*, 1996, pp. 212–219.

[6] I. M. Tsai, S. Y. Kuo, and D. S. L. Wei, "Quantum boolean circuit approach for searching and unordered database," in *Proc. 2nd IEEE Conf. Nanotechnol.*, Aug. 2002, pp. 325–318.

[7] M. Oskin, F. T. Chong, I. L. Chuang, and J. Kubiatowicz, "Building quantum wires: The long and the short of it," in *Proc. 30th Annu. Int. Symp. Comput. Architecture*, 2003, pp. 374–385.

[8] C. H. Bennett, G. Brassard, C. Crépeau, R. Josza, A. Peres, and W. K. Wootters, "Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels," *Phys. Rev. Lett.*, vol. 70, pp. 1895–1899, 1993.

[9] E. Biham, B. Huttner, and T. Mor, "Quantum cryptographic network based on quantum memories," *Phys. Rev. A, Gen. Phys.*, vol. 54, pp. 2651–2658, Oct. 1996.

[10] C. H. Bennett, G. Brassard, and A. K. Ekert, "Quantum cryptography," *Scientific Amer.*, pp. 50–57, Oct. 1992.

[11] W. Dür, H. J. Briegel, J. I. Cirac, and P. Zoller, "Quantum repeater based on entanglement purification," *Phys. Rev. A, Gen. Phys.*, vol. 59, pp. 169–181, 1999.

[12] S. T. Cheng, C.-Y. Wang, and M. H. Tao, "Quantum communication for wireless wide-area networks," *IEEE J. Sel. Areas Commun.*, pp. 1424–1432, Jul. 2005.

[13] P. Høyer, J. Neerbek, and Y. Shi, "Quantum complexities of ordered searching, sorting and element distinctness," in *Proc. 28th Int. Colloq. Automata, Languages Programming*, 2001, pp. 62–73.

[14] C. Moore and M. Nilsson, "Parallel Quantum Computation and Quantum Codes,", ArXive e-print quant-ph/9 808 027, 1998.

[15] A. Barenco, C. Bennett, R. Cleve, D. P. Divincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Phys. Rev. A, Gen. Phys.*, vol. 52, no. 5, pp. 3457–3467, 1995.

[16] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM F. Res. Dev.*, vol. 5, pp. 183–183, 1961.

[17] I. M. Tsai and S. Y. Kuo, "Digital switching in the quantum domain," *IEEE Trans. Nanotechnol.*, vol. 1, no. 5, pp. 154–164, Sep. 2002.

[18] I. M. Tsai, S. Y. Kuo, S. L. Huang, Y. C. Lin, and T. T. Chen, "Experimental Realization of an NMR Quantum Switch,", ArXive e-print quant-ph/0 405 170, 2004.

[19] M. K. Shukla, R. Ratan, and R. Y. Oruc, "A quantum self-routing packet switch," in *Proc. 38th Annu. Conf. Inf. Sciences Syst.*, Mar. 2004, pp. 484–489.

[20] D. Dieks, "Communication by EPR devices," *Phys. Lett. A*, vol. 92, no. 6, pp. 271–272, 1982.

[21] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, pp. 802–803, 1982.
[22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA: MIT Press, 2001.
[23] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Invariant Quantum Algorithms for Insertion into an Ordered List,", ArXive e-print quant-ph/9 901 059, 1999.
[24] A. Ambainis, "A better lower bound for quantum algorithms searching an ordered list," in *Proc. 40th Annu. IEEE Symp. Foundations Comput. Sci.*, 1999, pp. 352–357.
[25] A. J. Arul, "Impossibility of Comparing and Sorting Quantum States," ArXive, e-print quant-ph/0 107 085, 2001.
[26] H. Klauck, R. Spalek, and R. del Wolf, "Quantum and classical strong direct product theorems and optimal time-space tradeoffs," ArXive, e-print quant-ph/0402123, 2004.
[27] H. Klauck, "Quantum time-space tradeoffs for sorting," in *Proc. 35th ACM Symp. Theory Computing*, 2003, pp. 69–76.
[28] D. Nassimi and S. Sahni, "Parallel permutation and sorting algorithms and a new generalized connection network," *J. Assoc. Computing Mach.*, vol. 29, pp. 642–667, Jul. 1982.
[29] K. W. Cheng and C. C. Tseng, "Quantum full adder and subtractor," *Electron. Lett.*, vol. 38, pp. 1343–1344, Oct. 2002.

**Sheng-Tzong Cheng** received the B.S. and M.S. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, R.O.C., in 1985 and 1987, respectively, and the M.S. and Ph.D. degrees in computer science from the University of Maryland, College Park, in 1993 and 1995, respectively.

He was an Assistant Professor of computer science and information engineering with National Dong Hwa University, Hualien, Taiwan, in 1995, and became an Associate Professor in 1996. He is currently a Professor with the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan. His research interests are in design and performance analysis of mobile computing, wireless communications, multimedia, and real-time systems.

**Chun-Yen Wang** received the B.S. degree in mathematics from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 2001, where he is currently working toward the Ph.D. degree in computer science and information engineering.

His research interests include wireless communications, mobile computing, quantum computation, and quantum communications.