Quantum Information and Computation, Vol. 10, No. 9&10 (2010) 0872–0890

QUANTUM ADDITION CIRCUITS AND UNBOUNDED FAN-OUT

YASUHIRO TAKAHASHI[1], SEIICHIRO TANI[1,2], and NOBORU KUNIHIRO[3]
[1] *NTT Communication Science Laboratories, NTT Corporation*
3-1 Morinosato-Wakamiya, Atsugi, Kanagawa 243-0198, Japan
[2] *Quantum Computation and Information Project, ERATO-SORST, JST*
5-28-3 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan
[3] *Graduate School of Frontier Sciences, The University of Tokyo*
5-1-5 Kashiwanoha, Kashiwa, Chiba 277-8561, Japan

Received September 24, 2009
Revised July 20, 2010

We first show how to construct an $O(n)$-depth $O(n)$-size quantum circuit for addition of two n-bit binary numbers with no ancillary qubits. The exact size is $7n-6$, which is smaller than that of any other quantum circuit ever constructed for addition with no ancillary qubits. Using the circuit, we then propose a method for constructing an $O(d(n))$-depth $O(n)$-size quantum circuit for addition with $O(n/d(n))$ ancillary qubits for any $d(n) = \Omega(\log n)$. If we are allowed to use unbounded fan-out gates with length $O(n^{\varepsilon})$ for an arbitrary small positive constant ε, we can modify the method and construct an $O(e(n))$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits for any $e(n) = \Omega(\log^* n)$. In particular, these methods yield efficient circuits with depth $O(\log n)$ and with depth $O(\log^* n)$, respectively. We apply our circuits to constructing efficient quantum circuits for Shor's discrete logarithm algorithm.

Keywords: quantum circuits, addition, unbounded fan-out, Shor's discrete logarithm algorithm
Communicated by: R Jozsa & M Mosca

1 Introduction

Since Shor's discovery of quantum algorithms for factoring and discrete logarithm problems [1], many studies have investigated ways of constructing quantum circuits for the algorithms [2, 3, 4, 5, 6, 7]. The resulting circuits are important not only for implementing the algorithms on a quantum computer but also for understanding the computational power of small quantum circuits. These studies have shown that addition of two binary numbers is a key operation for constructing quantum circuits for Shor's algorithms.

We consider the problem of constructing quantum circuits for addition of two binary numbers with a reduction in overall complexity. The complexity measures of a quantum circuit are its size, depth, and qubit tally. In general size and depth are closely correlated to the number of time steps required, whereas the number of qubits closely approximates the memory requirements. It is crucial to investigate quantum circuit optimization in terms of various cost measures such as size, depth, and essential qubits, considering that a lot of research projects have extensively been examining not a few ways of implementing quantum computers. While the size and depth of a quantum circuit can be decreased through the application of efficient classical gates, there is no obvious way to apply these techniques to reduce the number of essential qubits. In this paper, we regard the number of qubits as a primary consideration.

An unbounded fan-out gate on $n+1$ qubits copies a classical source bit into n copies. In particular, the gate on two qubits is a CNOT gate. If unbounded fan-out gates are available, sublogarithmic-depth quantum circuits for various operations can be constructed [8, 9]. Several studies [10, 11, 12, 13, 14] demonstrated the possibility of building the gate on a quantum computer using the techniques such as ion traps. This raises an important question: How much can we reduce the number of target qubits in a circuit without increasing overall complexity of the circuit in terms of size and depth? When we use unbounded fan-out gates, we consider the complexity measures (size, depth, and the number of qubits) for the number of target qubits of the gate. We call the number of target qubits the length of an unbounded fan-out gate.

There have been many studies of efficient quantum circuits for addition of two n-bit binary numbers. These circuits can be classified according to complexity with respect to depth. Draper's and Takahashi and Kunihiro's circuits have depth $O(n)$ and use no ancillary qubits [15, 16]. Takahashi and Kunihiro's is more efficient than Draper's since the sizes of Takahashi and Kunihiro's and Draper's are $O(n)$ and $O(n^2)$, respectively. Draper et al.'s and Takahashi and Kunihiro's circuits have depth $O(\log n)$ [17, 18]. Draper et al.'s uses $O(n)$ ancillary qubits and its size is $O(n)$. Takahashi and Kunihiro decreased the number of ancillary qubits to $O(n/\log n)$ without increasing the size asymptotically. Høyer and Špalek showed that, if unbounded fan-out gates with length $O(n)$ are available, an $O(\log^* n)$-depth circuit can be constructed [9]. They have not analyzed their circuit in terms of ancillary qubit tallies or overall size.

In this paper, we first show how to construct an $O(n)$-depth $O(n)$-size quantum circuit, based on the ripple-carry approach, with no ancillary qubits. The exact size is $7n-6$, which is smaller than that of any other quantum circuit ever constructed for addition without ancillary qubits. Moreover, the circuit is easier to implement than the previous circuits with no ancillary qubits in the sense that the circuit can be used directly on a linear nearest neighbor architecture [6], i.e., on a unidimensional array of qubits with nearest neighbor interactions only. By combining the circuit with the carry-lookahead approach, we then propose a method for constructing an $O(d(n))$-depth $O(n)$-size quantum circuit for addition with $O(n/d(n))$ ancillary qubits for any $d(n) = \Omega(\log n)$. The method is a generalized and simplified version of Takahashi and Kunihiro's method for constructing a logarithmic-depth circuit with a small number of qubits [18]. In particular, for $d(n) = \log n$, our method yields an $O(\log n)$-depth $O(n)$-size circuit with $O(n/\log n)$ ancillary qubits. The count of ancillary qubits in the purposed circuit is identical to that of Takahashi and Kunihiro's (TK) circuit while the overall complexity in terms of size is less than half of the TK circuit.

If we are allowed to use unbounded fan-out gates with length $O(n^\varepsilon)$ for an arbitrary small positive constant ε, we can modify our method and construct an $O(e(n))$-depth $O(n)$-size circuit with $O(n \log^{**} n/e(n))$ ancillary qubits for any $e(n) = \Omega(\log^* n)$, where $\log^{**} n$ is a slowly-growing function satisfying $\log^{**} n = o(\log^* n)$. The main point of this modification is to decrease the depth of the carry-lookahead part of our method by using a quantum version of Chandra et al.'s constant-depth classical circuit for addition with unbounded fan-in and fan-out gates [19]. In order to construct the quantum version, we require a quantum gate corresponding to an unbounded fan-in gate. We use Høyer and Špalek's small-depth quantum circuit for a generalized Toffoli operation with unbounded fan-out gates [9] as the gate. In particular, for $e(n) = \log^* n$, the modified method yields an $O(\log^* n)$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits. Though Høyer and Špalek have constructed an $O(\log^* n)$-depth circuit for addition as mentioned above, our construction differs in so far as the number of ancillary qubits, overall size, and length of an unbounded fan-out gate can be minimized concurrently.

This construction demonstrates that unbounded fan-out gates with a small length are sufficient to construct a sublogarithmic-depth circuit. For example, if we are allowed to use unbounded fan-out gates with length $O(\log n)$, we can construct an $O(\log n/\log\log n)$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits. Such a sublogarithmic-depth circuit cannot be constructed by using a quantum circuit only with gates on a bounded number of qubits [20] or by using a classical circuit only with bounded fan-in and unbounded fan-out gates [21].

Using our circuits for addition, we construct efficient quantum circuits for Shor's discrete logarithm algorithm for elliptic curves over the prime field $\mathrm{GF}(p)$. This is done by simply using our addition circuits in Proos and Zalka's circuit for Shor's discrete logarithm algorithm [5]. Since Proos and Zalka's circuit uses n ancillary qubits during addition, the use of our circuit with no ancillary qubits decreases the n ancillary qubits without increasing the original depth or size asymptotically, where n is the length of the binary representation for p. Moreover, we decrease the depth asymptotically by adding $o(n)$ ancillary qubits. Proos and Zalka's circuit with our addition circuits is more efficient than with the previous ones described above.

In contrast to the previous methods for constructing efficient quantum circuits for addition [15, 16, 17, 18, 9], our method is general in the sense that it can yield various types of efficient quantum circuits for addition. The greater level of generality and abstraction allows us to construct a variety of quantum circuits for various applications, a few of which we will explore here. For example, if we want to constrain the number of qubits, we can obtain a qubit-efficient circuit by setting $d(n) = n$ in our method. We can decrease the depth by setting $d(n) = \log n$. Moreover, we can choose an "intermediate" circuit by setting $d(n) = \sqrt{n}$.

2 Circuit with Depth $O(n)$

2.1 Ripple-Carry Approach

We use the standard notation for quantum states and the standard diagrams for quantum circuits [22]. As described, the measures of complexity of a quantum circuit are the associated size, depth, and number of qubits. While the relationship between the number of qubits and the circuit complexity is straightforward, other complexity metrics such as depth and size are less straightforward. The size of a circuit is defined as the total number of elementary gates in it. The elementary gates are one-qubit unitary gates, CNOT gates, controlled-R_t gates, and Toffoli gates, where $R_t|x\rangle = e^{2\pi i x/2^t}|x\rangle$ for $t \geq 1$ and $x \in \{0,1\}$. In Section 4, we use the gate for an unbounded fan-out operation F_t as an elementary gate, where F_t (on $t+1$ qubits) is defined as

$$F_t\left(|y\rangle \bigotimes_{i=0}^{t-1} |x_i\rangle\right) = |y\rangle \bigotimes_{i=0}^{t-1} |x_i \oplus y\rangle$$

for $y, x_i \in \{0,1\}$. The symbol $\oplus$ denotes addition modulo 2. All the gates explicitly used in our circuits are classical in the sense that they map computational basis states to computational basis states. The depth of a circuit is defined as follows. Input qubits are considered to have

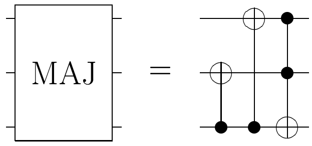

Fig. 1. The MAJ gate.

depth 0. For each gate G, the depth of G is equal to 1 plus the maximal depth of a gate on which G depends. The depth of a circuit is equal to the maximal depth of a gate in it. Intuitively, the depth is the number of layers in the circuit, where a layer consists of gates that can be performed simultaneously. A quantum circuit can use ancillary qubits, which start and end in the state $|0\rangle$. We usually count the number of ancillary qubits instead of the number of all qubits used in the circuit.

We consider the problem of constructing quantum circuits for the operation ADD_n defined as

$$\left(\bigotimes_{i=0}^{n-1} |b_i\rangle|a_i\rangle\right)|z\rangle \rightarrow \left(\bigotimes_{i=0}^{n-1} |s_i\rangle|a_i\rangle\right)|z \oplus s_n\rangle,$$

where $a_{n-1}\cdots a_0$ and $b_{n-1}\cdots b_0$ are the input binary numbers, $z \in \{0,1\}$, and $s_n \cdots s_0$ is the sum of the input binary numbers. Our linear-depth circuit and most of the previous ones with a small number of qubits are based on the ripple-carry approach. To explain the approach, we define the carry bit c_i ($0 \le i \le n$) as follows:

$$c_i = \begin{cases} 0 & i = 0, \\ \mathrm{MAJ}(a_{i-1}, b_{i-1}, c_{i-1}) & 1 \le i \le n, \end{cases}$$

where MAJ is the majority function for three bits defined as $\mathrm{MAJ}(a,b,c) = ab \oplus bc \oplus ca$. In the ripple-carry approach, the first step is to compute the carry bit c_1 by using a_0 and b_0 and c_0. Then, c_2 is computed by using a_1 and b_1 and c_1. This procedure is repeated until all carry bits are computed. After that, s_i ($0 \le i \le n$) is computed by the relationship

$$s_i = \begin{cases} a_i \oplus b_i \oplus c_i & 0 \le i \le n-1, \\ c_n & i = n. \end{cases}$$

When the ripple-carry approach is used, the key issue for constructing a quantum circuit with a small number of qubits is how to store carry bits. Cuccaro et al.'s circuits, which are based on the approach, use one ancillary qubit to store $c_0 = 0$ [23]. The carry bit c_i is stored in the qubit initially storing a_{i-1} for $1 \le i \le n$. To do this, they defined the gate for MAJ depicted in Fig. 1, which is the main component of their circuits. The gate maps $|c_i\rangle|b_i\rangle|a_i\rangle$ to $|c_i \oplus a_i\rangle|b_i \oplus a_i\rangle|c_{i+1}\rangle$. Takahashi and Kunihiro's circuit, which is also based on the ripple-carry approach, uses no ancillary qubits [16]. All the carry bits are stored in the qubit initially storing z. The main component of their circuit is also the MAJ gate. They use the property that the gate maps $|z \oplus b_i\rangle|z \oplus a_i\rangle|z \oplus c_i\rangle$ to $|b_i \oplus c_i\rangle|a_i \oplus c_i\rangle|z \oplus c_{i+1}\rangle$.

2.2 Our Circuit

We store the carry bit c_i in the qubit initially storing a_i for $0 \le i \le n-1$ and store the high-order bit c_n in the qubit initially storing z. This would be difficult to do if we use the

MAJ gate directly. Our idea is to divide the MAJ gate into two parts. The first part consists of two CNOT gates and the second one consists of one Toffoli gate. It is easy to verify that a Toffoli gate maps $|b_i \oplus a_i\rangle|a_i \oplus c_i\rangle|a_{i+1} \oplus a_i\rangle$ to $|b_i \oplus a_i\rangle|a_i \oplus c_i\rangle|a_{i+1} \oplus c_{i+1}\rangle$ for $1 \le i \le n-1$, where we consider a_n as z. Thus, using CNOT gates (the first parts of the MAJ gate) and a Toffoli gate, we first prepare the state

$$|b_1 \oplus a_1\rangle|a_1 \oplus c_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus a_i\rangle|a_i \oplus a_{i-1}\rangle \right) |z \oplus a_{n-1}\rangle.$$

By applying Toffoli gates (the second parts of the MAJ gate), we can compute c_i and store it in the qubit initially storing a_i. The final Toffoli gate computes c_n and stores it in the qubit initially storing z. The detailed construction is described below.

Let A_i and B_i denote the memory locations initially storing a_i and b_i, respectively, for $0 \le i \le n-1$. Let A_n be the memory location initially storing z. Location A_i $(0 \le i \le n-1)$ will store a_i, B_i $(0 \le i \le n-1)$ will store s_i, and A_n will store $z \oplus s_n$ at the end of the computation. Our circuit is constructed in the following six steps.

1. For $i = 1, \ldots, n-1$:

   Apply a CNOT gate to a pair of memory locations B_i and A_i where A_i is used for the control qubit.

2. For $i = n-1, \ldots, 1$:

   Apply a CNOT gate to a pair of memory locations A_i and A_{i+1} where A_i is used for the control qubit.

3. For $i = 0, \ldots, n-1$:

   Apply a Toffoli gate to a tuple of memory locations B_i, A_i and A_{i+1}, where B_i and A_i are used for the control qubit.

4. For $i = n-1, \ldots, 1$:

   Apply a CNOT gate to a pair of memory locations B_i and A_i where A_i is used for the control qubit. Then, apply a Toffoli gate to a tuple of memory locations B_{i-1}, A_{i-1} and A_i, where B_{i-1} and A_{i-1} are used for the control qubit.

5. For $i = 1, \ldots, n-2$:

   Apply a CNOT gate to a pair of memory locations A_i and A_{i+1} where A_i is used for the control qubit.

6. For $i = 0, \ldots, n-1$:

   Apply a CNOT gate to a pair of memory locations B_i and A_i where A_i is used for the control qubit.

The circuit for ADD_5 is depicted in Fig. 2.

We describe the changes of the input state of ADD_n to show that the circuit works correctly. In Step 1, the input state is transformed into

$$|b_0\rangle|a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus a_i\rangle|a_i\rangle \right) |z\rangle.$$

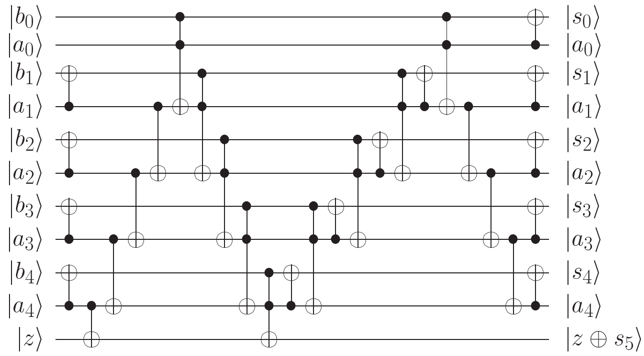

Fig. 2. The circuit for ADD_5.

In Step 2, the state is transformed into

$$|b_0\rangle|a_0\rangle|b_1 \oplus a_1\rangle|a_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus a_i\rangle|a_i \oplus a_{i-1}\rangle \right) |z \oplus a_{n-1}\rangle.$$

The first Toffoli gate in Step 3 transforms the state into

$$|b_0\rangle|a_0\rangle|b_1 \oplus a_1\rangle|a_1 \oplus c_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus a_i\rangle|a_i \oplus a_{i-1}\rangle \right) |z \oplus a_{n-1}\rangle.$$

This is repeated by using a Toffoli gate. The state after Step 3 is

$$|b_0\rangle|a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus a_i\rangle|a_i \oplus c_i\rangle \right) |z \oplus s_n\rangle. \tag{1}$$

In Step 4, the state is transformed into

$$|b_0\rangle|a_0\rangle|b_1 \oplus c_1\rangle|a_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus c_i\rangle|a_i \oplus a_{i-1}\rangle \right) |z \oplus s_n\rangle.$$

In Step 5, the state is transformed into

$$|b_0\rangle|a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus c_i\rangle|a_i\rangle \right) |z \oplus s_n\rangle.$$

Since $s_i = a_i \oplus b_i \oplus c_i$ for $0 \leq i \leq n-1$, the final step gives us the desired output state.

2.3 Complexity Analysis

From the construction, clearly our circuit makes no use of ancillary qubits. We compute the depth and size of the circuit for $n \geq 3$ precisely. In Step 1, the number of CNOT gates is $n-1$ and these gates can be performed simultaneously. Thus, the depth and size of Step 1 are 1 and $n-1$, respectively. In Step 2, the number of CNOT gates is $n-1$ and thus the depth and size of Step 2 are $n-1$. In Step 3, the number of Toffoli gates is n and thus the depth

Table 1. Comparison of Our Circuit and Previous Circuits

Circuit	Ancilla	Size	Toffoli	Depth	LNN
Cuccaro et al. [23]	1	$6n+1$	$2n$	$6n+1$	$\surd$
Cuccaro et al. [23]	1	$9n-8$	$2n-1$	$2n+4$	$\surd$
Draper [15]	0	$1.5n^2+4.5n+2$	0	$5n+3$	—
Takahashi et al. [16]	0	$10n-9$	$4n-5$	$8n-7$	—
Our Circuit	0	$7n-6$	$2n-1$	$5n-3$	$\surd$

and size of Step 3 are n. In Step 4, the number of CNOT gates is $n-1$ and the number of Toffoli gates is $n-1$. Thus, the depth and size of Step 4 are $2n-2$. In Step 5, the number of CNOT gates is $n-2$ and thus the depth and size of Step 5 are $n-2$. In Step 6, the number of CNOT gates is n and these gates can be performed simultaneously. Thus, the depth and size of Step 6 are 1 and n, respectively. Thus, the depth and size of the whole circuit are $5n-3$ and $7n-6$, respectively. The numbers of CNOT and Toffoli gates are $5n-5$ and $2n-1$, respectively.

As discussed in [6], many proposed quantum computer architectures deal with a unidimensional array of qubits with nearest neighbor interactions only. Thus, it is important for a circuit to work on such a linear nearest neighbor (LNN) architecture. When the input and output binary numbers are arranged on an LNN architecture in an interleaved manner (as in Fig. 2), our circuit can be used directly on an LNN architecture in the sense that the circuit can be transformed into one on an LNN architecture without increasing the size or depth asymptotically.

A comparison of our circuit and the previous ones with a small number of qubits is summarized in Table 1. The symbol "$\surd$" in the LNN column means that the circuit can be used directly on an LNN architecture in the sense described above. The symbol "—" means that we do not know whether this is the case for the circuit. The size of our circuit is less than that of any other quantum circuit ever constructed for ADD_n with no ancillary qubits. When we regard the number of qubits as a primary consideration, our circuit is more efficient than the previous circuits in Table 1.

Although there exists a size-efficient or depth-efficient circuit with a single ancillary qubit [23], it is worth noting that the difference between the total number of ancillary qubits used by parallel applications of our circuit (as in the next section) and that of previous circuits with a single ancillary qubit depends on the number of circuits applied in parallel and may become large. Moreover, since Toffoli gates are on three qubits and thus may be harder to implement than the other gates (on a smaller number of qubits), it is worth noting that the number of Toffoli gates in our circuit is $2n-1$, which is less than or equal to those of the previous circuits in Table 1 (excluding Draper's $O(n^2)$-size circuit).

3 General Method

3.1 Combination Method

The ripple-carry approach decreases the number of ancillary qubits but requires a greater depth. The carry-lookahead approach decreases the depth but requires many qubits [17]. Our method is based on the combination of these methods and is a generalized and simplified version of Takahashi and Kunihiro's method for constructing a logarithmic-depth circuit with a small number of qubits [18]. In this section, we review the previous method. The carry-lookahead approach is described by using two bits $p[i,j]$ $(1 \le i < j \le n)$ and $g[i,j]$ $(0 \le i < j \le n)$ [17]. The bit $p[i,j]$ is 1 if a carry bit is propagated but not generated from bit position i to bit position j, and $g[i,j]$ is 1 if a carry bit is generated between bit positions i and j. The $p[i,j]$ and $g[i,j]$ are computed by the following relations:

- For any i such that $1 \le i \le n-1$, $p[i,i+1] = a_i \oplus b_i$.
- For any i,j such that $1 \le i < i+1 < j \le n$, $p[i,j] = p[i,t]p[t,j]$ for any t satisfying $i < t < j$.
- For any i such that $0 \le i \le n-1$, $g[i,i+1] = a_i b_i$.
- For any i,j such that $0 \le i < i+1 < j \le n$, $g[i,j] = g[i,t]p[t,j] \oplus g[t,j]$ for any t satisfying $i < t < j$.

We note that $p[i,j]$ and $g[i,j]$ are mutually exclusive, that is, $p[i,j]g[i,j] = 0$ for all $i < j$ and thus $p[i,j] \oplus g[i,j] = p[i,j] \vee g[i,j]$. It holds that $g[0,j] = c_j$ for all $1 \le j \le n$.

Draper et al.'s quantum carry-lookahead adder first computes $p[i,i+1]$ $(1 \le i \le n-1)$ and $g[i,i+1]$ $(0 \le i \le n-1)$. Then, it computes $g[0,i]$ $(1 \le i \le n)$ by successively doubling the sizes of the intervals under consideration. Lastly, it computes s_i $(0 \le i \le n)$, where $s_0 = p[0,1]$, $s_i = p[i,i+1] \oplus g[0,i]$ $(1 \le i \le n-1)$, and $s_n = g[0,n]$. The key circuit is the one for the second step. We call this circuit the CARRY$_1$ gate. In general, the CARRY$_l$ gate is a circuit for the operation

$$\bigotimes_{i=1}^{\lfloor n/2^{l-1} \rfloor -1} |p_{l-1}[i]\rangle \bigotimes_{j=0}^{\lfloor n/2^{l-1} \rfloor -1} |g_{l-1}[j]\rangle \to \bigotimes_{i=1}^{\lfloor n/2^{l-1} \rfloor -1} |p_{l-1}[i]\rangle \bigotimes_{j=0}^{\lfloor n/2^{l-1} \rfloor -1} |g[0, 2^{l-1}(j+1)]\rangle, \quad (2)$$

where $1 \le l \le \lfloor \log n \rfloor - 1$, $p_{l-1}[i] = p[2^{l-1}i, 2^{l-1}(i+1)]$, and $g_{l-1}[i] = g[2^{l-1}i, 2^{l-1}(i+1)]$ [18]. The CARRY$_l$ gate uses $\sum_{t=l}^{\lfloor \log n \rfloor -1} (\lfloor n/2^t \rfloor - 1)$ ancillary qubits and its depth and size are $O(\log n - l)$ and $O(\sum_{t=l}^{\lfloor \log n \rfloor -1} (\lfloor n/2^t \rfloor - 1))$, respectively. Draper et al.'s quantum carry-lookahead adder uses $O(n)$ ancillary qubits and its depth and size are $O(\log n)$ and $O(n)$, respectively.

In Takahashi and Kunihiro's combination method, the input binary number $a_{n-1} \cdots a_0$ is divided into n/k blocks of length k, where we assume that n is a power of two for simplicity and set $k = 2^{\lfloor \log \log n \rfloor}$ and $l = \lfloor \log \log n \rfloor + 1$. Note that $k = \Theta(\log n)$ and n is divisible by k. That is, we consider a k-bit binary number $a(j) = a_{(j+1)k-1} \cdots a_{jk}$ for $0 \le j \le n/k - 1$. Similarly, we consider $b(j)$ for $b_{n-1} \cdots b_0$. Roughly speaking, the previous method is described as follows:

1. Compute the high-order bit of $a(j) + b(j)$, which is $g_{l-1}[j] = g[jk, (j+1)k]$, using the ripple-carry approach [16] for $0 \le j \le n/k - 1$.
2. Compute the value $\bigwedge_{i=0}^{k-1} (a_{jk+i} \oplus b_{jk+i})$, which is $p_{l-1}[j] = p[jk, (j+1)k]$, using Barenco et al.'s circuit for a generalized Toffoli operation T_k [24] for $0 \le j \le n/k - 1$, where T_t (on $t+1$ qubits) is defined as

$$T_t \left(|y\rangle \bigotimes_{i=0}^{t-1} |x_i\rangle \right) = |y \oplus \bigwedge_{i=0}^{t-1} x_i\rangle \bigotimes_{i=0}^{t-1} |x_i\rangle.$$

3. Compute the carry bit $c_{jk} = g[0, jk]$ using the values computed in Steps 1 and 2 for $1 \le j \le n/k$. This is done by using the CARRY_l gate.

4. Compute the carry bit $g[0, i]$ using the carry bits computed in Step 3 for $1 \le i \le n$ and obtain s_i for $0 \le i \le n$. This is done by a circuit based on the ripple-carry approach as in Step 1.

The whole circuit uses $O(n/k)$ $(= O(n/\log n))$ ancillary qubits and its depth and size are $O(k)$ $(= O(\log n))$ and $O(n)$, respectively.

3.2 Our Method

Our idea is to divide the input binary numbers into $n/d(n)$ blocks of length $d(n)$ in Takahashi and Kunihiro's method, where $d(n) = \Omega(\log n)$. By using the $\mathrm{CARRY}_{\log d(n)+1}$ gate, we can construct an $O(d(n))$-depth $O(n)$-size circuit with $O(n/d(n))$ ancillary qubits. This is a simple generalization of the previous method. Though this allows us to construct an $O(d(n))$-depth circuit for any $d(n) = \Omega(\log n)$ in contrast to the previous method, it, of course, does not improve the previous $O(\log n)$-depth circuit.

To obtain an efficient circuit, we simplify Steps 1, 2, and 4 in the previous method using the circuit for addition in Section 2. The simplification of Step 4 is due to a direct application of the circuit for addition. To simplify Steps 1 and 2, we use only the first halves of our circuit for addition and Barenco et al.'s circuit for T_n [24]. The first half of the circuit for addition outputs the high-order bit of $a(j)+b(j)$ and appropriate inputs to Barenco et al.'s circuit. We use only the first half and we can thus save Toffoli gates, but some qubits represent unuseful values. An important point is that Barenco et al.'s circuit can use these qubits as uninitialized ancillary qubits. We use the first half of Barenco et al.'s circuit and we can thus again save Toffoli gates, but some qubits have unuseful values. This is not a problem since these qubits are reset to the initial values in later steps. The details are described below.

To simplify Steps 1 and 2, since we need to compute only the two bits $g[i, j]$ and $p[i, j]$ for some i, j, it suffices to construct an efficient quantum circuit for the operation

$$\left(\bigotimes_{i=0}^{w-1} |b_i\rangle|a_i\rangle\right)|0\rangle|0\rangle \to \left(\bigotimes_{i=0}^{w-1} |p[i, i+1]\rangle|r_i\rangle\right)|g[0, w]\rangle|p[0, w]\rangle,$$

where $a_{w-1}\cdots a_0$ and $b_{w-1}\cdots b_0$ are the input binary numbers, $r_0 = a_0$, and $r_i = a_i \oplus g[0, i] \oplus p[0, i]$ $(1 \le i \le w-1)$. Let A_i and B_i denote the memory locations initially storing a_i and b_i, respectively. Let G and P be the memory locations initially storing 0. Location A_i will store r_i, B_i will store $p[i, i+1]$, G will store $g[0, w]$, and P will store $p[0, w]$ at the end of the computation. The circuit is defined as follows:

1. Apply Steps 1, 2, and 3 of the circuit (for two w-bit binary numbers) in Section 2 to a tuple of memory locations A_i $(0 \le i \le w-1)$ and B_i $(0 \le i \le w-1)$ and G to obtain the state (1) with $n = w$ and $z = 0$.

2. Apply a CNOT gate to a pair of memory locations A_0 and B_0, where A_0 is used for the control bit.

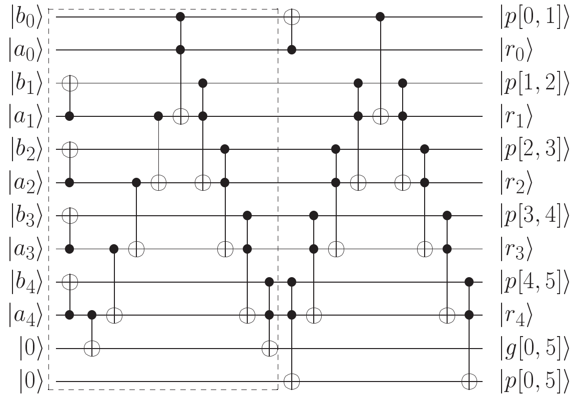

Fig. 3. The INIT_5 gate. A dashed-line box represents the part for computing $g[0,5]$, which is the first half of our circuit for addition in Section 2.

3. Apply the first half of Barenco et al.'s circuit for T_w to a tuple of memory locations A_i $(0 \leq i \leq w-1)$ and B_i $(0 \leq i \leq w-1)$ and P, where A_i is used as an uninitialized ancillary memory location.

Step 1 writes the value $s_w = g[0,w]$ into the memory location G. The memory location A_i stores the value $r_i \oplus p[0,i]$ and the memory location B_i stores the value $p[i,i+1]$ for all $i > 0$. Step 2 writes the value $p[0,1]$ into the memory location B_0 and thus the memory location B_i stores the value $p[i,i+1]$ for all i. The first half of Barenco et al.'s circuit outputs $p[0,1] \wedge \cdots \wedge p[w-1,w] = p[0,w]$ without changing the value in the memory location B_i. Step 3 writes the value $p[0,w]$ into the memory location P and writes the value r_i into the memory location A_i. The whole circuit uses no ancillary qubits and its depth and size are $O(w)$. We call the circuit the INIT_w gate. The INIT_5 gate is depicted in Fig. 3.

To simplify Step 4, it suffices to construct an efficient quantum circuit for the operation

$$\left(|c\rangle \bigotimes_{i=0}^{w-1} |b_i\rangle|a_i\rangle\right) \rightarrow \left(|c\rangle \bigotimes_{i=0}^{w-1} |t_i\rangle|a_i\rangle\right),$$

where $c \in \{0,1\}$, $a_{w-1}\cdots a_0$ and $b_{w-1}\cdots b_0$ are the input binary numbers, $t_j = a_j \oplus b_j \oplus d_j$ $(0 \leq j \leq w-1)$, and d_j is defined as

$$d_j = \begin{cases} c & j = 0, \\ \mathrm{MAJ}(a_{j-1}, b_{j-1}, d_{j-1}) & 1 \leq j \leq w-1. \end{cases}$$

We can directly apply the circuit in Section 2 to constructing such a circuit and thus omit the details. The circuit uses no ancillary qubits and its depth and size are $O(w)$. We call the circuit the SUM_w gate. The SUM_5 gate is depicted in Fig. 4.

3.3 The Whole Circuit

We construct a quantum circuit for ADD_n. For simplicity, we assume that n is a power of two. Let $d(n) = \Omega(\log n)$. We set $k = 2^{\lfloor \log d(n) \rfloor}$ and $l = \lfloor \log d(n) \rfloor + 1$. Note that $k = \Theta(d(n))$ and n is divisible by k. As described in Section 3.1, we consider k-bit binary numbers $a(j)$ and

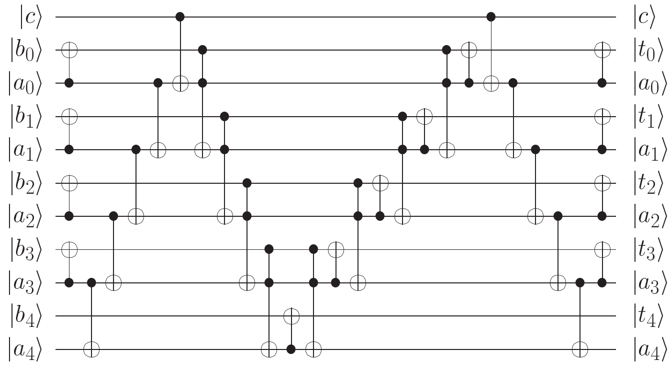

Fig. 4. The SUM_5 gate.

$b(j)$. Let A_i and B_i denote the memory locations initially storing a_i and b_i, respectively. Let Z be the memory location initially storing $z \in \{0,1\}$. Location A_i will store a_i, B_i will store s_i, and Z will store $z \oplus s_n$ at the end of the computation. We assume that there are ancillary memory locations initially storing 0. The first half of our circuit is defined as follows:

1. Apply the INIT_k gate to memory locations storing $a(j)$ and $b(j)$ and to two ancillary memory locations storing 0 for $0 \le j \le n/k-1$. The gate writes $g_{l-1}[j]$ and $p_{l-1}[j]$ into the ancillary memory locations.

2. Apply the CARRY_l gate to memory locations storing all $g_{l-1}[j]$ and all $p_{l-1}[j]$ and to ancillary memory locations storing 0. The gate writes $c_{(j+1)k}$ into the memory location storing $g_{l-1}[j]$ for $0 \le j \le n/k-1$.

3. Apply the gates in Step 1 in reverse order, where we exclude the gates applied to memory locations storing $c_{(j+1)k}$ for $0 \le j \le n/k-1$. These gates reset the whole state to the initial state except that the value $c_{(j+1)k}$ is stored in the memory location as described in Step 2 for $0 \le j \le n/k-1$.

4. Apply the SUM_k gate to memory locations storing $a(j+1)$ and $b(j+1)$ and to a memory location storing $c_{k(j+1)}$ to obtain $s_{k(j+1)}, \ldots, s_{k(j+2)-1}$ for $0 \le j \le n/k-2$. Apply a simplified gate of the SUM_k gate to memory locations storing $a(0)$ and $b(0)$ to obtain $s_0, \ldots, s_{k-1}$. The whole state is the desired output state except that the value $c_{(j+1)k}$ is stored in the memory location as described in Step 2 for $0 \le j \le n/k-1$.

The last half part deletes unnecessary carry bits (computed in Step 2) using the fact that the carry bits generated for computing $a+s'$ are the same as those for computing $a+b$, where s' is the bitwise complement of s [17].

5. Apply a NOT gate to B_i to write $s_i \oplus 1$ into B_i for $0 \le i \le n-k-1$.

6. Apply the gates of Step 3 in reverse order, then apply the gates of Step 2 in reverse order, then apply the gates of Step 1 in reverse order, where we exclude the gates applied to memory locations storing $a(n/k-1)$ and $b(n/k-1)$ since we do not erase the last carry bit. The gate writes 0 into a memory location storing $c_{k(j+1)}$ for $0 \le j \le n/k-1$.

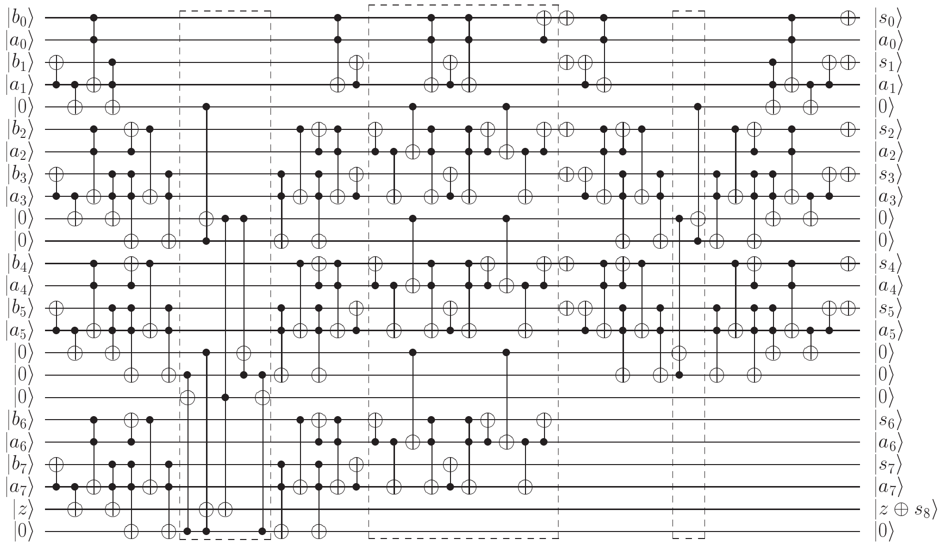

Fig. 5. The circuit for ADD_8, where $d(n) = \log n$. The first and third dashed-line boxes represent the carry-lookahead part [17, 18]. The second one represents the parallel applications of the SUM_2 gate.

7. Apply a NOT gate to B_i to write s_i into B_i for $0 \le i \le n-k-1$.

The whole circuit for $d(n) = \log n$ and $n = 8$ (and thus $k = l = 2$) is depicted in Fig. 5.

We compute the number of ancillary qubits, the depth, and the size precisely. For simplicity, we count only Toffoli gates as in [17, 18]. Step 1 requires $\frac{2n}{k}$ ancillary qubits to use $\frac{n}{k}$ INIT_k gates. The gate consists of $3n-2$ Toffoli gates for $n \ge 3$. Thus, the depth and size of Step 1 are $3k - O(1)$ and $3n - O(n/k)$, respectively. The CARRY_l gate in Step 2 uses $\frac{n}{k} - O(\log n)$ ancillary qubits and its depth and size are $2\log\frac{n}{k} + O(1)$ and $\frac{4n}{k} + O(\log n)$, respectively, where $\frac{n}{k} \ge 4$ [18]. Step 3 is the same as Step 1. Step 4 uses $\frac{n}{k}$ SUM_k gates. The gate consists of $2n-2$ Toffoli gates for $n \ge 3$. Thus, the depth and size of Step 4 are $2k - O(1)$ and $2n - O(n/k)$, respectively. The other steps are the same as the above steps excluding Step 4. Our circuit uses $\frac{3n}{k} - O(\log n)$ ancillary qubits and its depth and size are $14k + 4\log\frac{n}{k} + O(1)$ and $14n - O(n/k)$, respectively, where $\frac{n}{k} \ge 4$. Thus, the circuit uses $O(n/d(n))$ ancillary qubits and its depth and size are $O(d(n))$ and $O(n)$, respectively. For example, for $d(n) = \log n$ and $n \ge 16$, the number of ancillary qubits, the depth, and the size are approximately $3n/\log n$, $18\log n$, and $14n$, respectively. The corresponding previous bounds are $3n/\log n$, $30\log n$, and $29n$. That is, in this case, the number of ancillary qubits in our circuit is the same as that in Takahashi and Kunihiro's [18] and the leading coefficient of the expression of the size in our circuit is less than half that in Takahashi and Kunihiro's.

4 Circuit with Depth $o(\log n)$

4.1 Chandra et al.'s Classical Circuit

If we use only one-qubit and two-qubit gates as elementary gates, we cannot construct an $o(\log n)$-depth circuit for ADD_n. This is simply shown by using the logarithmic lower bound for the depth of the circuit for F_n [20]. To construct an $o(\log n)$-depth circuit, we decrease the depth of the carry-lookahead part of our method in Section 3 by using a quantum version of Chandra et al.'s efficient classical circuit for addition with (classical) unbounded fan-out gates [19]. We assume that we have unbounded fan-out gates (described in Section 2) as elementary gates. We first consider the simple case where we have unbounded fan-out gates with a long length and then reduce the length.

Chandra et al.'s method for constructing the circuit is a generalization of the carry-lookahead approach. Besides the (classical) unbounded fan-out gates, the circuit uses unbounded fan-in gates that compute logical AND (or OR) of an unbounded number of input bits. The depth and size of the circuit for two m-bit binary numbers are $O(1)$ and $O(m\log^{**} m)$, respectively, where

$$\log^{**} t = \min\{j | \overbrace{\log^* \cdots \log^*}^{j} t \leq 1\},\ \log^* t = \min\{j | \overbrace{\log \cdots \log}^{j} t \leq 1\}.$$

It can be shown that $\log^{**} m = o(\log^* m)$. Though the definition of the depth of a classical circuit is similar to that of a quantum circuit, the definition of the size of a classical circuit in [19] is different from that of a quantum circuit. More precisely, a classical circuit is defined as a directed acyclic graph and the size is the number of edges in the circuit and the depth is the length of a longest path from an input node to an output node. Chandra et al. give a tighter bound on the size of the circuit, but we use the above bound since it is sufficient for showing that our circuits in Sections 4.2 and 4.3 use a sublinear number of ancillary qubits.

4.2 Simple Case

4.2.1 Quantum Version of Chandra et al.'s Circuit

We transform Chandra et al.'s classical circuit for two m-bit binary numbers into its quantum version. Since the size (that is, the number of edges) of the circuit is $O(m\log^{**} m)$, it suffices to consider an unbounded fan-out gate with length $O(m\log^{**} m)$ and a T_t gate (corresponding to an unbounded fan-in gate with t inputs in the classical circuit) with $t = O(m\log^{**} m)$. We assume that we have unbounded fan-out gates with length $O(m\log^{**} m)$. If we have one-qubit gates, CNOT gates, T_t gates, and unbounded fan-out gates with length $O(m\log^{**} m)$, Chandra et al.'s classical circuit can be simply transformed into its quantum version. Note that an OR gate in Chandra et al.'s circuit is transformed into a T_t gate with NOT gates. However, in our setting, we have only one-qubit gates, CNOT gates, and unbounded fan-out gates with length $O(m\log^{**} m)$. Thus, we require a quantum circuit for T_t (consisting of one-qubit gates, CNOT gates, and unbounded fan-out gates with length $O(m\log^{**} m)$). We use Høyer and Špalek's circuit for the T_t operation (defined in Section 3.1) as the T_t gate [9]. They showed that, if unbounded fan-out gates with length $O(t)$ are available, an $O(\log^* t)$-depth $O(t)$-size quantum circuit for T_t can be constructed. We can show that Høyer and Špalek's circuit uses $O(t)$ ancillary qubits. Since we have unbounded fan-out gates with length $O(m\log^{**} m)$, we can directly use Høyer and Špalek's circuit for T_t with $t = O(m\log^{**} m)$. Thus, we obtain a quantum version of Chandra et al.'s circuit. We call the circuit the GCLA$_m$ circuit, which stands for the generalized carry-lookahead approach for two m-bit binary numbers.

The complexity of the GCLA$_m$ circuit is analyzed as follows. To compute the depth of the circuit, since the depth of the original circuit is $O(1)$, it suffices to consider a T_{t_1} gate, where t_1 is the maximum number of inputs of T_t gates in the GCLA$_m$ circuit. The depth of the T_{t_1} gate is $O(\log^* t_1)$. Since $t_1 = O(m \log^{**} m)$, the depth of the T_{t_1} gate is $O(\log^*(m \log^{**} m))$. Since $O(\log^*(m \log^{**} m)) \subseteq O(\log^*(2^m)) \subseteq O(1 + \log^* m) \subseteq O(\log^* m)$, the depth of the GCLA$_m$ circuit is $O(\log^* m)$. To compute the size of the circuit, we define A_t as the number of unbounded fan-in gates with t inputs in Chandra et al.'s circuit, which is equal to the number of T_t gates in the GCLA$_m$ circuit. Since the size of Chandra et al.'s circuit is $O(m \log^{**} m)$, $\sum_t tA_t = O(m \log^{**} m)$. The size of a T_t gate is $O(t)$. The number of the other gates in the GCLA$_m$ circuit is $O(m \log^{**} m)$ (and the size of each gate is 1). Thus, the size of the GCLA$_m$ circuit is $O(\sum_t tA_t + m \log^{**} m) = O(m \log^{**} m)$. A similar argument shows that the number of ancillary qubits in the GCLA$_m$ circuit is $O(m \log^{**} m)$. That is, the GCLA$_m$ circuit uses $O(m \log^{**} m)$ ancillary qubits and its depth and size are $O(\log^* m)$ and $O(m \log^{**} m)$, respectively.

4.2.2 Modification of Our Method

We modify our method in Section 3.3 by using the GCLA$_m$ circuit as the CARRY$_l$ gate. Let $e(n) = \Omega(\log^* n)$. We set k and l as in Section 3.3 but with $e(n)$ in place of $d(n)$. Note that $k = 2^{l-1} = \Theta(e(n))$. We assume that we are allowed to use unbounded fan-out gates with length $O(n)$. Chandra et al.'s circuit for two $\lfloor n/2^{l-1} \rfloor$-bit binary numbers is directly applied to perform the operation (2). Thus, we set $m = \lfloor n/2^{l-1} \rfloor$. In this case, $O(m \log^{**} m) = O(n(\log^{**}(n/2^{l-1}))/2^{l-1})$, which is bounded by $O(n)$. Since we have unbounded fan-out gates with length $O(n)$, we can use the complexity analysis described in Section 4.2.1. The GCLA$_m$ circuit, which is the CARRY$_l$ gate, uses $O(n(\log^{**}(n/2^{l-1}))/2^{l-1})$ ancillary qubits and its depth and size are $O(\log^*(n/2^{l-1}))$ and $O(n(\log^{**}(n/2^{l-1}))/2^{l-1})$, respectively. For simplicity, we consider slightly weaker bounds for the number of ancillary qubits and size; it uses $O(n(\log^{**} n)/2^{l-1})$ ancillary qubits and its size is $O(n(\log^{**} n)/2^{l-1})$.

The complexity of the whole circuit obtained by the modified method is analyzed as in the original method. Step 1 uses $O(n/k)$ ancillary qubits and its depth and size are $O(k)$ and $O(n)$, respectively. Step 2 uses $O(n(\log^{**} n)/k)$ ancillary qubits and its depth and size are $O(\log^*(n/k))$ and $O(n(\log^{**} n)/k)$, respectively. Step 4 requires no new ancillary qubits and its depth and size are $O(k)$ and $O(n)$, respectively. The other steps are similar to the above steps. Thus, the whole circuit uses $O(n(\log^{**} n)/e(n))$ $(= o(n))$ ancillary qubits and its depth and size are $O(e(n))$ and $O(n)$, respectively. In particular, for $e(n) = \log^* n$, the modified method yields an $O(\log^* n)$-depth $O(n)$-size circuit with $O(n(\log^{**} n)/\log^* n)$ $(= o(n))$ ancillary qubits.

4.3 Reduction of the Length of an Unbounded Fan-Out Gate

We prove that the length of an unbounded fan-out gate can be restricted to $O(n^\varepsilon)$ in the modified method without increasing the complexity of the circuit, where ε is any small positive constant. Suppose that we are allowed to use unbounded fan-out gates with length $f(n)$. An unbounded fan-out gate with length $t = O(m \log^{**} m)$ (and $m = \lfloor n/2^{l-1} \rfloor$) can be simply simulated by using an $O(\log t/\log f(n) + 1)$-depth $O(t/f(n) + 1)$-size circuit with no ancillary qubits that consists only of unbounded fan-out gates with length $f(n)$. In the following, using this simulation, we reconsider the complexity of the T_t gate, the GCLA$_m$ circuit, and the circuit our method in Section 4.2 yields.

4.3.1 T_t gate

The T_t gate, which is Høyer and Špalek's circuit for the T_t operation, is constructed as follows:

1. Construct an $O(1)$-depth $O(t \log t)$-size circuit with $O(t \log t)$ ancillary qubits for reducing the computation of OR of t bits to that of $O(\log t)$ bits.

2. Using the circuit in Step 1, for any $d > 0$, construct an $O(d + \log^* t)$-depth $O(dt \log^{(d)} t)$-size circuit for T_t with $O(dt \log^{(d)} t)$ ancillary qubits, where $\log^{(d)} t$ is the d-times iterated logarithm $\log \cdots \log t$.

3. Using the circuit in Step 2, construct an $O(\log^* t)$-depth $O(t)$-size circuit for T_t with $O(t)$ ancillary qubits.

We can modify the above steps using unbounded fan-out gates with length $f(n)$ as follows:

1. Construct an $O(\log t/ \log f(n) + 1)$-depth $O(t \log t)$-size circuit with $O(t \log t)$ ancillary qubits for reducing the computation of OR of t bits to that of $O(\log t)$ bits.

2. Using the circuit in Step 1, for any $d > 0$, construct an $O(d + \log^* t + \log t/ \log f(n) + d \log \log t/ \log f(n))$-depth $O(dt \log^{(d)} t)$-size circuit for T_t with $O(dt \log^{(d)} t)$ ancillary qubits.

3. Using the circuit in Step 2, construct an $O(\log t/ \log f(n) + \log^* t)$-depth $O(t)$-size circuit for T_t with $O(t)$ ancillary qubits.

To see this, we first analyze Step 1 in Høyer and Špalek's construction. In this step, an unbounded fan-out gate with length $O(\log t)$ is used in parallel to make $O(\log t)$ copies of each of the t input bits. Moreover, an unbounded fan-out gate with length $O(t)$ is used in parallel to prepare appropriate ancillary qubits $O(\log t)$ times. As described above, an unbounded fan-out gate with length $O(\log t)$ can be simulated by using an $O(\log \log t/ \log f(n) + 1)$-depth $O(\log t/f(n) + 1)$-size circuit with no ancillary qubits. Similarly, an unbounded fan-out gate with length $O(t)$ can be simulated by using an $O(\log t/ \log f(n) + 1)$-depth $O(t/f(n) + 1)$-size circuit. Thus, the depth of the T_t gate is $O(\log t/ \log f(n) + 1)$. The size is $O(t \cdot (\log t/f(n) + 1) + (\log t) \cdot (t/f(n) + 1)) = O(t \log t)$. These simulations do not require any ancillary qubits. That is, in Step 1, the number of ancillary qubits and size remain unchanged even if we consider unbounded fan-out gates with length $f(n)$. Thus, they also do so in Steps 2 and 3. Step 2 of Høyer and Špalek's construction is done by using Step 1 $O(\log^* t)$ times to reduce the computation of OR of t bits to that of a constant number of bits. Step 3 is done by reducing the computation of OR of t bits to that of $t/ \log^* t$ bits and by using Step 2 with $d = \log^* t$. These procedures can be simply applied to the case where we use unbounded fan-out gates with length $f(n)$ and imply the desired depth bound.

4.3.2 The GCLA$_m$ circuit

To compute the depth of the GCLA$_m$ circuit, it suffices to consider a T_{t_1} gate for some t_1 and an unbounded fan-out gate with some length t_2. The depth of the T_{t_1} gate is $O(\log t_1/ \log f(n) + \log^* t_1)$ and the depth of an unbounded fan-out gate with length t_2 is $O(\log t_2/ \log f(n) + 1)$. Since t_1 and t_2 cannot be greater than the size of Chandra et al.'s circuit, the depth of the GCLA$_m$ circuit is $O(\log m/ \log f(n) + \log^* m)$. To compute the size, we define B_t as the number of unbounded fan-out gates with length t used (implicitly) in Chandra et al.'s original circuit, which is equal to the number of unbounded fan-out gates with length t (that are not used in T_s gates for any s) in the GCLA_m circuit. Since the size of Chandra et al.'s circuit is $O(m \log^{**} m)$, $\sum_t tB_t = O(m \log^{**} m)$. If $t \geq f(n)$, an unbounded fan-out gate with length t can be simulated by an $O(t/f(n))$-size circuit. Thus, the size related to unbounded fan-out gates with length greater than or equal to $f(n)$ in the GCLA_m circuit (that is, $\sum_{t \geq f(n)}(t/f(n))B_t$) is $O(m \log^{**} m)$ since $\sum_t tB_t = O(m \log^{**} m)$. The size related to the T_t gates (that is, $O(\sum_t tA_t)$) is $O(m \log^{**} m)$. The number of the other gates is $O(m \log^{**} m)$ (and the size of each gate is 1). Thus, the size of the GCLA_m circuit is $O(m \log^{**} m)$. The number of ancillary qubits is the same as the size. That is, the GCLA_m circuit uses $O(m \log^{**} m)$ ancillary qubits and its depth and size are $O(\log m/\log f(n) + \log^* m)$ and $O(m \log^{**} m)$, respectively. Since $m = \lfloor n/2^{l-1} \rfloor$, the circuit uses $O(n(\log^{**}(n/2^{l-1}))/2^{l-1})$ ancillary qubits and its depth and size are $O(\log(n/2^{l-1})/\log f(n) + \log^*(n/2^{l-1}))$ and $O(n(\log^{**}(n/2^{l-1}))/2^{l-1})$, respectively. For simplicity, we consider slightly weaker bounds; it uses $O(n(\log^{**} n)/2^{l-1})$ ancillary qubits and its depth and size are $O(\log n/\log f(n) + \log^*(n/2^{l-1}))$ and $O(n(\log^{**} n)/2^{l-1})$, respectively.

4.3.3 Our Circuit

We set $f(n) = n^{\varepsilon}$ and use the GCLA_m circuit as the CARRY_l gate, where ε is any small positive constant. In this case, the CARRY_l gate uses $O(n(\log^{**} n)/2^{l-1})$ ancillary qubits and its depth and size are $O(\log^*(n/2^{l-1}))$ and $O(n(\log^{**} n)/2^{l-1})$, respectively. This is the same situation as that in Section 4.2 except that the length of an unbounded fan-out gate in the CARRY_l gate is at most n^{ε}. Thus, the whole circuit uses $O(n(\log^{**} n)/e(n))$ $(= o(n))$ ancillary qubits and its depth and size are $O(e(n))$ and $O(n)$, respectively. If we set $e(n) = \log^* n$, we obtain an $O(\log^* n)$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits.

It is worth noting that the above method for constructing a circuit for ADD_n yields an $o(\log n)$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits using unbounded fan-out gates with a small length. For example, we set $f(n) = \log n$ and $e(n) = \log n/\log\log n$. In this case, the CARRY_l gate uses $O(n \log^{**} n \log\log n/\log n)$ ancillary qubits and its depth and size are $O(\log n/\log\log n)$ and $O(n \log^{**} n \log\log n/\log n)$, respectively. This yields an $O(\log n/\log\log n)$-depth $O(n)$-size circuit with $O(n \log^{**} n \log\log n/\log n)$ ancillary qubits. Note that, though the size of the CARRY_l gate is $o(n)$, the size of the whole circuit is $O(n)$ since the other parts of the circuit uses $O(n)$ gates. Such an $o(\log n)$-depth circuit cannot be constructed by using a quantum circuit only with gates on a bounded number of qubits [20] or by using a classical circuit only with bounded fan-in and unbounded fan-out gates [21]. Hence, unbounded fan-out gates even with a small length are useful for constructing efficient quantum circuits for addition.

5 Application

We consider the prime field $\mathrm{GF}(p)$ for some prime $p > 3$. An elliptic curve E over $\mathrm{GF}(p)$ is the set of points $(x, y) \in \mathrm{GF}(p) \times \mathrm{GF}(p)$ satisfying $y^2 = x^3 + ax + b$, where the constants $a, b \in \mathrm{GF}(p)$ and $4a^3 + 27b^2 \neq 0$, together with the point at infinity $\mathcal{O}$. It is known that the addition operation in E can be defined and that E with the addition operation forms an abelian group with $\mathcal{O}$ serving as its identity [25]. Let $P \in E$, $\langle P \rangle$ be the subgroup of E generated by P, and $|\langle P \rangle|$ be the order of $\langle P \rangle$. The discrete logarithm problem over the elliptic curve E with respect to the base P is defined as follows: Given a point $Q \in \langle P \rangle$, find the integer $0 \leq d \leq |\langle P \rangle| - 1$ such that $Q = dP$. Shor's discrete logarithm algorithm solves the problem in time polynomial in the length of the binary representation for $|\langle P \rangle|$ with high probability [1]. As in [5], we assume that the length of the binary representation for $|\langle P \rangle|$ is equal to that of the binary representation for p.

Proos and Zalka constructed an efficient quantum circuit for Shor's discrete logarithm algorithm for elliptic curves over $\mathrm{GF}(p)$ [5]. Let n be the length of the binary representation for p. The depth and size of the circuit are $O(n^3)$. The dominant cost is $O(n^2)$ applications of an $O(n)$-depth $O(n)$-size quantum circuit for ADD_n with n ancillary qubits. For counting the number of qubits in the circuit, it suffices to count the number of qubits in the circuit for division in $\mathrm{GF}(p)$ that maps $|x\rangle|y\rangle$ to $|x\rangle|y/x\rangle$ for $x\ (\neq 0), y \in \mathrm{GF}(p)$. The circuit for division in $\mathrm{GF}(p)$ uses about $5n$ qubits: $2n$ qubits are used for the input register and about $3n$ qubits are used in the circuit for the extended Euclidean algorithm. In the circuit for the extended Euclidean algorithm, about $2n$ qubits are used for the input binary numbers and intermediate results, and n qubits are used for ancillary qubits during ADD_n.

By simply replacing Proos and Zalka's circuit for ADD_n with our circuit in Section 2, we can eliminate the n ancillary qubits during ADD_n since our circuit for ADD_n does not use any ancillary qubits. The resulting circuit uses about $4n$ qubits. Since Proos and Zalka do not describe the precise depth or size of their circuit for ADD_n, we cannot compare the depth or size of the resulting circuit with that of the original one precisely. However, the depth and size of our circuit for ADD_n are asymptotically the same as those of Proos and Zalka's. Thus, the depth and size of the resulting circuit are asymptotically the same as those of the original circuit.

By adding $o(n)$ ancillary qubits to the circuit obtained above, we can decrease the depth asymptotically. As shown in Section 3, for any $d(n) = \Omega(\log n)$, we have an $O(d(n))$-depth $O(n)$-size circuit for ADD_n with $O(n/d(n))$ ancillary qubits. If we use this circuit as above, we obtain $O(n^2 d(n))$-depth $O(n^3)$-size circuit for Shor's discrete logarithm algorithm with $4n + O(n/d(n))$ qubits. Moreover, as shown in Section 4, if we are allowed to use unbounded fan-out gates with length $O(n^\varepsilon)$ for an arbitrary small positive constant ε, we have an $O(e(n))$-depth $O(n)$-size circuit for ADD_n with $o(n)$ ancillary qubits for any $e(n) = \Omega(\log^* n)$. This circuit yields an $O(n^2 e(n))$-depth $O(n^3)$-size circuit for Shor's discrete logarithm algorithm with $4n + o(n)$ qubits. We can also use the previous circuits for ADD_n to improve Proos and Zalka's circuit. However, they do not yield more efficient quantum circuits for Shor's discrete logarithm algorithm than our circuit described above. This is simply because our circuit for ADD_n is more efficient than the previous ones.

6 Conclusions and Future Work

We constructed an $O(n)$-depth $O(n)$-size quantum circuit for ADD_n with no ancillary qubits. The size is less than that of any other quantum circuit ever constructed for ADD_n with no ancillary qubits. Using the circuit, we proposed a method for constructing a small-size quantum circuit for ADD_n with a small number of qubits that has a given depth. In particular, we showed that, if we are allowed to use unbounded fan-out gates with length $O(n^\varepsilon)$ for an arbitrary small positive constant ε, we can construct an $O(\log^* n)$-depth $O(n)$-size circuit with $o(n)$ ancillary qubits. We applied our circuits to constructing efficient quantum circuits for Shor's discrete logarithm algorithm.

Interesting challenges would be to find ways of improving the quantum circuits described in this paper. For example, can we construct an $O(\log n)$-depth $O(n)$-size quantum circuit for ADD_n with $O(1)$ ancillary qubits? Can we construct an $O(1)$-depth $O(n)$-size quantum circuit for ADD_n with $O(n)$ ancillary qubits using unbounded fan-out gates? In the classical case, we cannot construct an $O(1)$-depth $O(n)$-size (that is, the number of edges) circuit for addition with unbounded fan-in and fan-out gates [26].

Acknowledgments

The authors thank Yasuhito Kawano, Go Kato, and the anonymous referees for their valuable comments.

References

1. P. W. Shor (1994), *Algorithms for quantum computation: discrete logarithms and factoring*, In Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, pages 124–134.
2. V. Vedral, A. Barenco, and A. Ekert (1996), *Quantum networks for elementary arithmetic operations*, Phys. Rev. A, 54(1):147–153.
3. C. Zalka (1998), *Fast versions of Shor's quantum factoring algorithm*, quant-ph/9806084.
4. S. Beauregard (2003), *Circuit for Shor's algorithm using* $2n + 3$ *qubits*, Quantum Information and Computation, 3(2):175–185.
5. J. Proos and C. Zalka (2003), *Shor's discrete logarithm quantum algorithm for elliptic curves*, Quantum Information and Computation, 3(4):317–344.
6. A. G. Fowler, S. J. Devitt, and L. C. L. Hollenberg (2004), *Implementation of Shor's algorithm on a linear nearest neighbour qubit array*, Quantum Information and Computation, 4(4):237–251.
7. Y. Takahashi and N. Kunihiro (2006), *A quantum circuit for Shor's factoring algorithm using* $2n + 2$ *qubits*, Quantum Information and Computation, 6(2):184–192.
8. F. Green, S. Homer, C. Moore, and C. Pollett (2002), *Counting, fanout, and the complexity of quantum ACC*, Quantum Information and Computation, 2(1):35–65.
9. P. Høyer and R. Špalek (2005), *Quantum fan-out is powerful*, Theory of Computing, 1(5):81–103.
10. J. I. Cirac and P. Zoller (1995), *Quantum computations with cold trapped ions*, Phys. Rev. Lett., 74(20):4091–4094.
11. K. Mølmer and A. Sørenson (1999), *Multiparticle entanglement of hot trapped ions*, Phys. Rev. Lett., 82(9):1835–1838.
12. X. Wang, A. Sørenson, and K. Mølmer (2001), *Multibit gates for quantum computing*, Phys. Rev. Lett., 86(17):3907–3910.
13. S. A. Fenner (2003), *Implementing the fanout gate by a Hamiltonian*, quant-ph/0309163.
14. S. A. Fenner and Y. Zhang (2004), *Implementing fanout, parity, and Mod gates via spin exchange interactions*, quant-ph/0407125.
15. T. G. Draper (2000), *Addition on a quantum computer*, quant-ph/0008033.
16. Y. Takahashi and N. Kunihiro (2005), *A linear-size quantum circuit for addition with no ancillary qubits*, Quantum Information and Computation, 5(6):440–448.
17. T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore (2006), *A logarithmic-depth quantum carry-lookahead adder*, Quantum Information and Computation, 6(4&5):351–369.
18. Y. Takahashi and N. Kunihiro (2008), *A fast quantum circuit for addition with few qubits*, Quantum Information and Computation, 8(6&7):636–649.

19. A. K. Chandra, S. Fortune, and R. Lipton (1983), *Unbounded fan-in circuits and associative functions*, In Proceedings of the 15th Annual ACM Symposium on Theory of Computing, pages 52–60.
20. M. Fang, S. Fenner, F. Green, S. Homer, and Y. Zhang (2006), *Quantum lower bounds for fanout*, Quantum Information and Computation, 6(1):46–57.
21. N. Pippenger (1987), *The complexity of computations by networks*, IBM Journal of Research and Development, 31(2):235–243.
22. M. A. Nielsen and I. L. Chuang (2000), *Quantum Computation and Quantum Information*, Cambridge University Press.
23. S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton (2005), *A new quantum ripple-carry addition circuit*, The Eighth Workshop on Quantum Information Processing. Also on quant-ph/0410184.
24. A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter (1995), *Elementary gates for quantum computation*, Phys. Rev. A, 52(5):3457–3467.
25. D. Hankerson, A. Menezes, and S. Vanstone (2003), *Guide to Elliptic Curve Cryptography*, Springer.
26. D. Dolev, C. Dwork, N. Pippenger, and A. Wigderson (1983), *Superconcentrators, generalizers and generalized connectors with limited depth*, In Proceedings of the 15th Annual ACM Symposium on Theory of Computing, pages 42–51.