entropy

IMPACT FACTOR 2.1

Indexed in: PubMed

CITESCORE 4.9

Article

Prepare Linear Distributions with Quantum Arithmetic Units

Junxu Li

Special Issue

Entropy in Classical and Quantum Information Theory with Applications

Edited by

Dr. Dominik Szczęśniak and Prof. Dr. Sabre Kais

MDPI

https://doi.org/10.3390/e26110912

entropy

MDPI

Article

Prepare Linear Distributions with Quantum Arithmetic Units

Junxu Li

Department of Physics, College of Science, Northeastern University, Shenyang 110819, China; lijunxu1996@gmail.com

Abstract: Quantum arithmetic logic units (QALUs) perform essential arithmetic operations within a quantum framework, serving as the building blocks for more complex computations and algorithms in quantum computing. In this paper, we present an approach to prepare linear probability distributions with quantum full adders. There are three main steps. Firstly, Hadamard gates are applied to the two input terms, preparing them at quantum states corresponding to uniform distribution. Next, the two input terms are summed up by applying quantum full adder, and the output sum is treated as a signed integer under two's complement representation. By the end, additional phase -1 is introduced to the negative components. Additionally, we can discard either the positive or negative components with the assistance of the Repeat-Until-Success process. Our work demonstrates a viable approach to prepare linear probability distributions with quantum adders. The resulting state can serve as an intermediate step for subsequent quantum operations.

Keywords: quantum computing; quantum adders; quantum arithmetic logic units (QALUs)

1. Introduction

Quantum computing signifies a transformative shift in the landscape of computational capabilities, which introduces a level of computational power that enables the efficient resolution of problems deemed intractable by classical computing paradigms [1–13]. Quantum arithmetic logic units (QALUs) constitute the centerstone of quantum computing. In classical computing, the elementary arithmetic logic units (ALUs) are designed to execute basic arithmetic operations. Similarly, a variety of QALUs, such as quantum adders, subtractors, and multipliers, serve as the building blocks for more complicated algorithms in quantum computing. The best known example is Shor's algorithm [14,15], where the quantum modular arithmetic plays a vital role.

Among various QALUs, the quantum adder plays a pivotal role, representing a fundamental operation integral to the execution of all algorithms that are constructed upon it. The first quantum adder was proposed in 1996, when Vedral and coworkers explicitly constructed the first quantum ripple carry adder [16], which is a significant landmark in the development of QALUs. Since then, a variety of quantum adders has been developed, including ripple carry [16–18], carry lookahead [19,20], carry save [21], and hybrid [22,23] structures. On the other hand, quantum adders can also be implemented based on quantum Fourier transform (QFT) [24,25]. QFT-based quantum adders [26,27] typically commence with a QFT block that converts the input state to the frequency domain, performs addition using controlled phase gates, and then reverts the state to the original domain through an inverse Fourier transform [28,29]. Moreover, recently, a novel approach was proposed to prepare arbitrary normal distributions in quantum registers, utilizing the QFT-based quantum adders [30].

In this paper, we revisit the fundamental quantum binary adders and propose a practical method for generating a linear probability distribution utilizing quantum adders. The following sections are organized as follows. In Section 2, we revisit the simple implementation of quantum binary adders, and demonstrate the outputs of iterative additions that sum up several standalone uniform superposition states. Next, in Section 3, we present

Citation: Li, J. Prepare Linear Distributions with Quantum Arithmetic Units. *Entropy* **2024**, *26*, 912. https://doi.org/10.3390/e26110912

Academic Editor: Rosario Lo Franco

Received: 23 September 2024
Revised: 24 October 2024
Accepted: 25 October 2024
Published: 28 October 2024

how to prepare the linear probability distribution with quantum full adders in the two's complement representation. By the end, the discussions and conclusions are present in Section 4.

2. Materials and Methods

Like classical binary adders, quantum binary adders add up two given terms, and return the output sum. In Figure 1, we depict the diagram of a typical quantum binary adder. The box colored in blue represents a quantum binary adder, and the inputs are given on the left side, whereas the outputs on the right side. There are $2N$ qubits initialized at state $|a_{N-1}\cdots a_0\rangle$ and $|b_{N-1}\cdots b_0\rangle$, representing the input binary numbers $(a_{N-1}\cdots a_0)_{bin}$ and $(b_{N-1}\cdots b_0)_{bin}$. One qubit is initialized at state $|0\rangle$, representing the input carry (in general, the input carry is set as 0). There are N qubits all initialized at state $|0\rangle$ to store the output sum. Generally, the quantum binary adders do not change the state of the inputs, and the output sum, a $N+1$ digit binary number, is stored in the other $N+1$ qubits (including the qubit that represents the input carry). As depicted in Figure 1a, the $N+1$ qubits, which are prepared at $|0\rangle$ at the beginning, are converted to state $|s_N\cdots s_0\rangle$ after applying the quantum binary adder. The output state represents the output sum $(s_N\cdots s_0)_{bin}$, and we have

$$(s_N\cdots s_0)_{bin} = (a_{N-1}\cdots a_0)_{bin} + (b_{N-1}\cdots b_0)_{bin} \tag{1}$$

For clarity, here, subscript *bin* indicates that the number is binary.

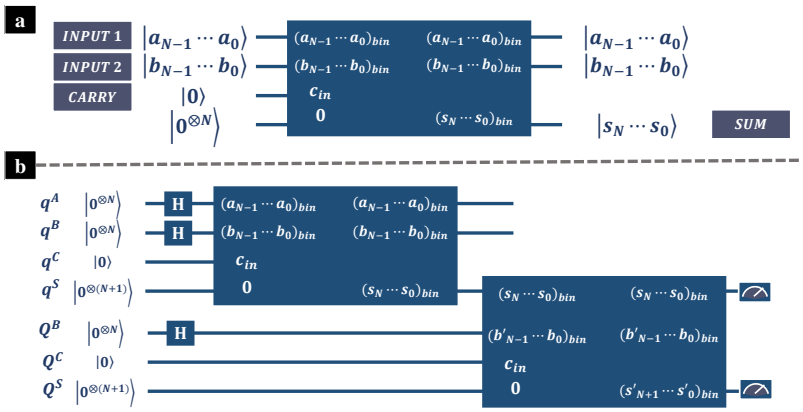

Figure 1. (**a**) Schematic diagram of quantum adder supporting multiple bits. The box colored in blue represents a quantum binary adder, and the inputs are given on the left side, whereas the outputs are on the right side. (**b**) Schematic diagram of summing up three input terms with two quantum adders. At the beginning, all qubits are initialized at state $|0\rangle$, and the inputs are prepared at uniform superposition by applying Hadamard gates. Qubits denoted as q^A, q^B, q^C, q^S are involved in the first addition. The output sum is then sent to the succeeding addition, along with qubits Q^B, Q^C, and Q^S. The superscripts indicate that the qubits represent the input terms (A, B), the input carry (C), or the output sum (S).

In classical electronics, a cascade of one-bit full adders constitutes the simplest adder supporting multiple bits. The one-bit full adder adds three inputs [31]: two binary digits denoted as a and b, along with an input carry denoted as c_{in}. Then, the one-bit full adder produces two outputs: the sum s and output carry c_{out}. The one-bit full adder performs binary addition as

$$s = a \oplus b \oplus c_{in} \tag{2}$$

$$c_{out} = (a \cdot b) + (c_{in} \cdot (a \oplus b)) \tag{3}$$

where $a, b, c_{in}, c_{out}, s = 0, 1$, and $a + b + c_{in} = s + 2c_{out}$. Similarly, the quantum one-bit full adder, denoted as U_+, performs binary addition as

$$U_+|a, b, c_{in}, 0\rangle = |a, b, s, c_{out}\rangle \tag{4}$$

where the outputs s, c_{out} are as given in Equations (2) and (3). In Figure 2b, we present the circuit of a typical quantum one-bit full adder, which is formed by two Toffoli gates along with three CNOT gates. There are four qubits involved in operation U_+, two qubits represent the input terms a and b, one qubit represents the input carry c_{in} and stores the output sum s, and the last qubit initialized at state $|0\rangle$ stores the output carry c_{out}.

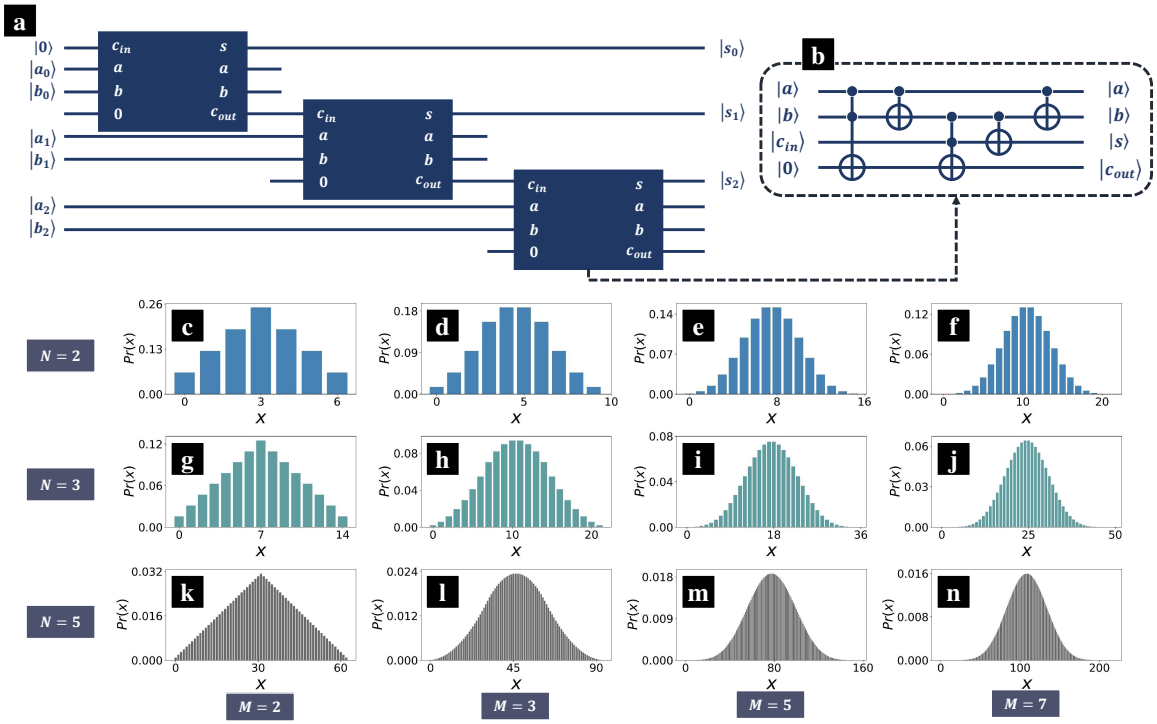

Figure 2. (**a**) Schematic diagram of summing up three input terms with two quantum adders. (**b**) A typical implementation of quantum one-bit full adder. There are in total three CNOT gates and two Toffoli gates. (**c**–**n**) The theoretical prediction of the output sum, where the input terms are M standalone uniform superpositions prepared by Hadamard gates. Each input term represents a N digit unsigned binary number. $Pr(x)$ is the probability of obtaining result x, where we denote the sum as an integer x, and the standard quantum state of the output sum corresponds to the binary form of x.

The simplest quantum adder that supports multiple bits can be formed by cascading multiple one-bit full adders. Figure 2a is a schematic diagram of three cascade quantum one-bit full adders, where the blue boxes indicate operation U_+, with inputs on the left side and outputs on the right side. The addition starts from the least significant bits (LSBs) a_0 and b_0, and the first operation U_+ is applied on the corresponding qubits. The output sum s_0 of the LSB addition is also the LSB of the output sum, whereas the output carry is sent to the addition of the next digit, a_1 and b_1. In brief, the output carry performs as the input carry in the addition of the next digit, and the output carry of the most significant bit (MSB) is just the MSB of the output sum.

In general, we can sum up M standalone input terms by applying $M-1$ quantum adders that support multiple bits. Figure 1a is a schematic diagram of summing up three input terms with two quantum adders. At the beginning, all qubits are initialized at state $|0\rangle$, and the inputs are prepared at uniform superposition by applying Hadamard gates. Qubits denoted as q^A, q^B, q^C, and q^S are involved in the first addition. The output sum is then sent to the succeeding addition, along with qubits Q^B, Q^C, and Q^S. The superscripts indicate that the qubits represent the input terms (A, B), the input carry (C), or the output

sum (S). After adding up M standalone input terms, the output quantum state, denoted as Φ_M, can be given as

$$\Phi_M = \frac{1}{\sqrt{2^{NM}}} \bigotimes_{m=1}^{M} \left(\sum_{a_m=0}^{2^N-1} |a_m\rangle \right) \otimes | \sum_{m=1}^{M} a_m \rangle \tag{5}$$

The M input terms are denoted as a_m, $m = 1, 2, \cdots, M$. As all inputs are N bit binary unsigned integers, a_m ranges from 0 to $2^N - 1$. Then, we measure the qubits that represent the output sum, and denote the probability of obtaining result x as $Pr(x)$, where for simplicity, we denote the sum as an integer x, and the standard quantum state of the output sum corresponds to the binary form of x. The theoretical prediction of $Pr(x)$ for various M and N are as depicted in Figure 2c–n, where there are in total M input terms, and each input is a N bit binary number. As N and M increase, the distribution $Pr(x)$ begins to resemble a normal distribution. Thus, it is a feasible approach to prepare normal distribution with quantum adders! [30] (In Ref. [30], Rattew and coworkers proposed a quantum algorithm for the efficient preparation of arbitrary normal distributions, utilizing the quantum random work, and quantum Fourier Transformations or quantum adders, along with Mid-Circuit Measurement and Reuse (MCMR) techniques. The quantum algorithm can be implemented with a polynomial-depth circuit. Later, in Section 3, we show that our approach can prepare normal distributions also with polynomial-depth quantum circuits.).

3. Results

Here, we concentrate on the case $M = 2$, where there are only two input terms. As both the inputs are N digit unsigned integers, the output sum is a $N + 1$ digit binary number. Recalling Equation (5), the overall quantum state after the addition (the input carry $c_{in} = 0$) can be written as

$$\Phi_{M=2} = \frac{1}{2^N} \sum_{a=0}^{2^N-1} \sum_{b=0}^{2^N-1} (|a\rangle \otimes |b\rangle \otimes |a+b\rangle) \tag{6}$$

where the two input terms are denoted as a and b, and we have $a, b \leq 2^N - 1$, whereas the output sum, for simplicity denoted as x, ranges from 0 to $2^{N+1} - 2$. Then, we measure the qubits corresponding to the sum after the addition, and denote $Pr(x)$ as the probability of obtaining result x. For $0 \leq x \leq 2^N - 1$, there are in total $x + 1$ pairs of inputs a, b that lead to output sum x: $a = 0, b = x$; $a = 1, b = x - 1$; $\cdots$ and $a = x, b = 0$. Similarly, for $2^N \leq x \leq 2^{N+1} - 2$, there are $2^{N+1} - x - 1$ pairs of inputs a, b that lead to output sum x: $a = 2^N - 1, b = x - 2^N + 1$; $a = 2^N - 2, b = x - 2^N + 2$; $\cdots$ and $a = x - 2^N + 1, b = 2^N - 1$. Therefore, the probability of obtaining result x can be given as

$$Pr(x|c_{in} = 0) = \begin{cases} \frac{1}{4^N}(x+1), & 0 \leq x \leq 2^N - 1 \\ \frac{1}{4^N}(2^{N+1} - x - 1), & 2^N \leq x \leq 2^{N+1} - 2 \end{cases} \tag{7}$$

$Pr(x)$ reaches to the peak at $x = 2^N - 1$, and $Pr(2^N - 1) = \frac{1}{2^N}$. The theoretical predictions of $Pr(x)$ with $M = 2$ are as depicted in Figure 2c,g,k, where the distribution $Pr(x)$ is shaped like an isosceles triangle.

Till now, the output sum has been treated as a $N + 1$ digit unsigned integer. Hereafter, we will demonstrate that the distribution $Pr(x)$ can be converted to a linear distribution with two's complement representation. Two's complement is the most common method of representing signed integers in classical computers [31,32]. In two's complement representation, the value χ of a $N + 1$ bit integer $(x_N, x_{N-1} \cdots x_0)_{bin}$ is given as

$$\chi = -2^N x_N + \sum_{j=0}^{N-1} 2^j x_j \tag{8}$$

where x_N is the sign bit indicating the sign, $x_N = 0$ for non-negative integers, and $x_N = 1$ for negative integers. The two's complement representation of a non-negative number is just its ordinary binary representation, with sign bit 0.

In two's complement representation, the $N+1$ digit binary number $(x_N x_{N-1} \cdots x_0)$ is treated as a signed integer. For $0 \leq x \leq 2^N - 1$, we have $x_N = 0$, and the readout χ is still x. On the other hand, for $2^N \leq x \leq 2^{N+1} - 2$, we have $x_N = 1$, and the readout χ is negative. Recalling that $x = \sum_{j=0}^{N} 2^N x_j$, where x is the value of the unsigned binary integer $(x_N \cdots x_0)_{bin}$, we have

$$\chi = x - 2^{N+1}, \qquad 2^N \leq x \leq 2^{N+1} - 2 \tag{9}$$

Thus, we can rewrite Equation (7) in two's complement representation,

$$Pr(\chi|c_{in} = 0) = \frac{1}{4^N}|\chi + 1|, \qquad -2^N \leq \chi \leq 2^N - 1 \tag{10}$$

Consider a simple example, the addition of two 3-digit unsigned binary numbers. In Table 1, we present the output sums for all possible inputs. For each input pairs, there are four components in Table 1. The first component (from top) is the output quantum state, where the input carry c_{in} is 0, and the second component is the corresponding readout of the sum (unsigned/signed). The other two components correspond to the case where $c_{in} = 1$, and will be discussed later. For example, consider the addition of $|100\rangle$ and $|101\rangle$. With input carry $c_{in} = 0$, the output quantum state is $|1001\rangle$. If we treat the 4-digit binary number $(1001)_{bin}$ as an unsigned integer, then the readout is $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 9$. However, if we treat $(1001)_{bin}$ as a signed integer in two's complement representation, then the readout will be $-1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = -7$.

Table 1. Table of the output sums, where both the input terms are 3-bit binary unsigned integers, and the input carry is set either 0 or 1. There are four components in each entry. From top to bottom, the first one is the output sum with input carry 0, and the second one is the corresponding readout (unsigned/signed), whereas the third component is the output sum with input carry 1, and the last one is the corresponding readout (unsigned/signed).

Inputs	$\|$**000**$\rangle$	$\|$**001**$\rangle$	$\|$**010**$\rangle$	$\|$**011**$\rangle$	$\|$**100**$\rangle$	$\|$**101**$\rangle$	$\|$**110**$\rangle$	$\|$**111**$\rangle$
$\|000\rangle$	$\|0000\rangle$ 0/0 $\|0001\rangle$ 1/1	$\|0001\rangle$ 1/1 $\|0010\rangle$ 2/2	$\|0010\rangle$ 2/2 $\|0011\rangle$ 3/3	$\|0011\rangle$ 3/3 $\|0100\rangle$ 4/4	$\|0100\rangle$ 4/4 $\|0101\rangle$ 5/5	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8
$\|001\rangle$	$\|0001\rangle$ 1/1 $\|0010\rangle$ 2/2	$\|0010\rangle$ 2/2 $\|0011\rangle$ 3/3	$\|0011\rangle$ 3/3 $\|0100\rangle$ 4/4	$\|0100\rangle$ 4/4 $\|0101\rangle$ 5/5	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7
$\|010\rangle$	$\|0010\rangle$ 2/2 $\|0011\rangle$ 3/3	$\|0011\rangle$ 3/3 $\|0100\rangle$ 4/4	$\|0100\rangle$ 4/4 $\|0101\rangle$ 5/5	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6
$\|011\rangle$	$\|0011\rangle$ 3/3 $\|0100\rangle$ 4/4	$\|0100\rangle$ 4/4 $\|0101\rangle$ 5/5	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6	$\|1010\rangle$ 10/−6 $\|1011\rangle$ 11/−5
$\|100\rangle$	$\|0100\rangle$ 4/4 $\|0101\rangle$ 5/5	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6	$\|1010\rangle$ 10/−6 $\|1011\rangle$ 11/−5	$\|1011\rangle$ 11/−5 $\|1100\rangle$ 12/−4

Table 1. *Cont.*

Inputs	$\|000\rangle$	$\|001\rangle$	$\|010\rangle$	$\|011\rangle$	$\|100\rangle$	$\|101\rangle$	$\|110\rangle$	$\|111\rangle$
$\|101\rangle$	$\|0101\rangle$ 5/5 $\|0110\rangle$ 6/6	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6	$\|1010\rangle$ 10/−6 $\|1011\rangle$ 11/−5	$\|1011\rangle$ 11/−5 $\|1100\rangle$ 12/−4	$\|1100\rangle$ 12/−4 $\|1101\rangle$ 13/−3
$\|110\rangle$	$\|0110\rangle$ 6/6 $\|0111\rangle$ 7/7	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6	$\|1010\rangle$ 10/−6 $\|1011\rangle$ 11/−5	$\|1011\rangle$ 11/−5 $\|1100\rangle$ 12/−4	$\|1100\rangle$ 12/−4 $\|1101\rangle$ 13/−3	$\|1101\rangle$ 13/−3 $\|1110\rangle$ 14/−2
$\|111\rangle$	$\|0111\rangle$ 7/7 $\|1000\rangle$ 8/−8	$\|1000\rangle$ 8/−8 $\|1001\rangle$ 9/−7	$\|1001\rangle$ 9/−7 $\|1010\rangle$ 10/−6	$\|1010\rangle$ 10/−6 $\|1011\rangle$ 11/−5	$\|1011\rangle$ 11/−5 $\|1100\rangle$ 12/−4	$\|1100\rangle$ 12/−4 $\|1101\rangle$ 13/−3	$\|1101\rangle$ 13/−3 $\|1110\rangle$ 14/−2	$\|1110\rangle$ 14/−2 $\|1111\rangle$ 15/−1

In Figure 3, a straightforward demonstration of the addition is present. The quantum circuit is as depicted in Figure 3a, where q^A and q^B represent the input terms, q^C represents the input carry, and the output sum is stored in q^S. Initially, all qubits are prepared at state $|0\rangle$. Hadamard gates are then applied on qubits q^A and q^B, preparing them at uniform superpositions as depicted in Figure 3c,d. Here, in $c_{in} = 0$, the operations in the dashed boxes are not applied. Then, the standard quantum adder is applied, summing the input terms, and the qubits q^S are measured. If we treat the output sum as an unsigned integer, then the distribution $Pr(x)$ is as depicted in Figure 3e, which is shaped as an isosceles triangle, similar to Figure 2c,g,k. On the other side, if we treat the output sum as a signed integer in two's complement representation, then we will obtain the distribution as shown in Figure 3g, which is a linear distribution according to Equation (10).

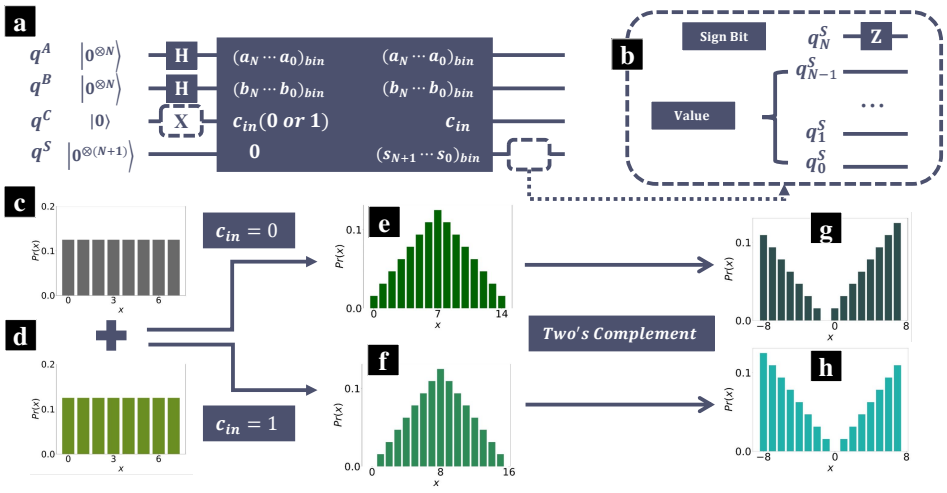

Figure 3. (**a**) Schematic diagram of the quantum circuit that sums up the two uniform superpositions, where q^A and q^B represent the input terms, q^C is the input carry, and the output sum is stored in q^S. The operations in dashed boxes are applied for $c_{in} = 1$. (**b**) To introduce phase -1 to the negative components, we can apply a Pauli-Z gate on q^S_N, which represents the sign bit of the output. (**c**,**d**) Visualization of the uniform superpositions after applying Hadamard gates. (**e**,**f**) Distribution of the probability of obtaining various results after measuring the output sums, which is treated as unsigned integers. (**g**,**h**) Distribution of the probability of obtaining various results after measuring the output sums, which is treated as signed integers in two's complement representation. In (**e**,**g**) we set the input carry $c_{in} = 0$, whereas in (**f**,**h**) $c_{in} = 1$.

Notice that in Figure 3g, the probability of obtaining result -1 is zero. Sometimes, we prefer to have the zero probability at 0, in other words, setting $Pr(\chi = 0) = 0$. For this case,

we can set the input carry $c_{in} = 1$. In this context, the sum is $a + b + 1$, where 1 is the input carry. Equation (7) can then be rewritten as

$$Pr(x|c_{in} = 0) = \begin{cases} \frac{1}{4^N}x, & 1 \le x \le 2^N - 1 \\ \frac{1}{4^N}(2^{N+1} - x), & 2^N \le x \le 2^{N+1} - 1 \end{cases} \tag{11}$$

The corresponding probability distribution is depicted in Figure 2f. If we treat the output as a signed integer in two's complement representation, we have

$$Pr(\chi|c_{in} = 0) = \frac{1}{4^N}|\chi|, \qquad -2^N \le \chi \le 2^N - 1 \tag{12}$$

The output sums for $c_{in} = 1$ are also given in Table 1, where the third component is the output sum with input carry 1, and the last one is the corresponding readout (unsigned/signed). In Figure 2h, the distribution of the probability of obtaining various results after measuring the output sums is depicted, where the output sum is treated as signed integers in two's complement representation. By this mean, the probability of obtaining result 0 is zero as shown in Figure 2h.

Although the probability is always non-negative, we can also introduce an additional phase to the negative components. As shown in Figure 2b, we can apply a Pauli-Z operation on the qubit q_N^S after applying the quantum adder. The qubit q_N^S represents the MSB of the output sum and is the sign bit if we treat the sum as a signed integer in two's complement representation. Recalling Equation (8), for negative components, the sign bit is 1; otherwise, the sign bit is 0. Thus, the Pauli-Z operation can introduce an additional phase -1 to the negative components.

Furthermore, we can also discard either the positive or negative components with the assistance of the Repeat-Until-Success process [33–35]. After applying the quantum adder, we can measure the qubit q_N^S. If we obtain result $|1\rangle$, then the distribution probability is $(c_{in} = 1)$

$$Pr_-(\chi|c_{in} = 1) = -\frac{1}{(1 + 2^N)2^{N-1}}\chi, \qquad -2^N \le \chi \le 0 \tag{13}$$

Otherwise, if we obtain result $|0\rangle$, the distribution probability is

$$Pr_+(\chi|c_{in} = 1) = \frac{1}{(2^N - 1)2^{N-1}}\chi, \qquad 0 \le \chi \le 2^N - 1 \tag{14}$$

where the subscripts $\pm$ indicate the measurement results. If we want to obtain a linear probability distribution for the negative range as shown in Equation (13), then we need to repeat the whole process until we obtain result $|1\rangle$ when measuring the qubit q_N^S. On the contrary, if we prefer to obtain the probability distribution for the positive range as shown in Equation (14), then we need to repeat until we obtain result $|0\rangle$.

The time complexity of the proposed approach is mainly determined by the quantum adder. Generally, the time complexity of a typical N bit binary quantum adder, as depicted in Figure 2b, is of the $\mathcal{O}(N)$ order, where there are $2N$ Toffoli gates along with $3N$ CNOT gates. However, due to the limited connectivity, it can be challenging to implement large quantities of Toffoli gates on the quantum computers in NISQ era. To address this issue, we propose an alternative design of quantum binary adders in our recent work [36], utilizing only Pauli-X gates, CNOT gates, and $C\sqrt{X}$ (CSX) gates, and all two-qubit gates are operated between nearest neighbor qubits. The time complexity of the quantum adder is still of the $\mathcal{O}(N)$ order. Therefore, to prepare the linear distribution as shown in Equation (12), the time complexity is $\mathcal{O}(N)$. Furthermore, the proposed approach can be extended to prepare normal distributions as depicted in Figure 2c–n. Theoretically, the time complexity of summing up M independent uniform superposition states is $\mathcal{O}((M-1)N)$. In other words, the proposed approach can be implementable with a polynomial-depth circuit.

4. Discussion

In this paper, we revisit the elementary quantum binary adders, and propose a viable approach to prepare linear probability distribution with the quantum adders. There are three main steps to obtain the linear probability distribution. Initially, Hadamard gates are applied to the input qubits, transforming them into uniform superposition states. Following this, the input qubits are processed through a quantum full adder, where their sum is computed, and the resulting output is interpreted as a signed integer in two's complement representation. At a subsequent stage, a phase shift of -1 is introduced to the components representing negative values. Furthermore, the positive or negative components may be selectively discarded using a Repeat-Until-Success protocol. The resulting output can serve as an intermediate state for succeeding quantum operations.

Funding: This research was funded by National Natural Science Foundation of China (NSFC) under Grant No. 12305012.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: All data and materials supporting the results or analyses presented in this paper are available from the corresponding author upon reasonable request.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Lloyd, S. Universal quantum simulators. *Science* **1996**, *273*, 1073–1078. [CrossRef] [PubMed]
2. Biamonte, J.; Wittek, P.; Pancotti, N.; Rebentrost, P.; Wiebe, N.; Lloyd, S. Quantum machine learning. *Nature* **2017**, *549*, 195–202. [CrossRef] [PubMed]
3. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
4. Li, Z.; Liu, X.; Wang, H.; Ashhab, S.; Cui, J.; Chen, H.; Peng, X.; Du, J. Quantum simulation of resonant transitions for solving the eigenproblem of an effective water Hamiltonian. *Phys. Rev. Lett.* **2019**, *122*, 090504. [CrossRef]
5. Cao, Y.; Romero, J.; Olson, J.P.; Degroote, M.; Johnson, P.D.; Kieferová, M.; Kivlichan, I.D.; Menke, T.; Peropadre, B.; Sawaya, N.P.; et al. Quantum chemistry in the age of quantum computing. *Chem. Rev.* **2019**, *119*, 10856–10915. [CrossRef]
6. Google AI Quantum and Collaborators; Arute, F.; Arya, K.; Babbush, R.; Bacon, D.; Bardin, J.C.; Barends, R.; Boixo, S.; Broughton, M.; Buckley, B.B.; et al. Hartree-Fock on a superconducting qubit quantum computer. *Science* **2020**, *369*, 1084–1089.
7. Li, J.; Kais, S. Quantum cluster algorithm for data classification. *Mater. Theory* **2021**, *5*, 6. [CrossRef]
8. Huang, H.Y.; Broughton, M.; Mohseni, M.; Babbush, R.; Boixo, S.; Neven, H.; McClean, J.R. Power of data in quantum machine learning. *Nat. Commun.* **2021**, *12*, 2631. [CrossRef]
9. Altman, E.; Brown, K.R.; Carleo, G.; Carr, L.D.; Demler, E.; Chin, C.; DeMarco, B.; Economou, S.E.; Eriksson, M.A.; Fu, K.M.C.; et al. Quantum simulators: Architectures and opportunities. *PRX Quantum* **2021**, *2*, 017003. [CrossRef]
10. Schlimgen, A.W.; Head-Marsden, K.; Sager, L.M.; Narang, P.; Mazziotti, D.A. Quantum simulation of open quantum systems using a unitary decomposition of operators. *Phys. Rev. Lett.* **2021**, *127*, 270503. [CrossRef]
11. Sajjan, M.; Li, J.; Selvarajan, R.; Sureshbabu, S.H.; Kale, S.S.; Gupta, R.; Singh, V.; Kais, S. Quantum machine learning for chemistry and physics. *Chem. Soc. Rev.* **2022**, *51*, 6475–6573. [CrossRef] [PubMed]
12. Li, J.; Gao, X.; Sajjan, M.; Su, J.H.; Li, Z.K.; Kais, S. Møller-Plesset Perturbation Theory Calculations on Quantum Devices. *arXiv* **2023**, arXiv:2308.01559.
13. Ma, H.; Liu, J.; Shang, H.; Fan, Y.; Li, Z.; Yang, J. Multiscale quantum algorithms for quantum chemistry. *Chem. Sci.* **2023**, *14*, 3190–3205. [CrossRef] [PubMed]
14. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
15. Ekert, A.; Jozsa, R. Quantum computation and Shor's factoring algorithm. *Rev. Mod. Phys.* **1996**, *68*, 733. [CrossRef]
16. Vedral, V.; Barenco, A.; Ekert, A. Quantum networks for elementary arithmetic operations. *Phys. Rev. A* **1996**, *54*, 147. [CrossRef]
17. Cuccaro, S.A.; Draper, T.G.; Kutin, S.A.; Moulton, D.P. A new quantum ripple-carry addition circuit. *arXiv* **2004**, arXiv:quant-ph/0410184.
18. Gidney, C. Halving the cost of quantum addition. *Quantum* **2018**, *2*, 74. [CrossRef]
19. Draper, T.G.; Kutin, S.A.; Rains, E.M.; Svore, K.M. A logarithmic-depth quantum carry-lookahead adder. *arXiv* **2004**, arXiv:quant-ph/0406142. [CrossRef]
20. Wang, S.; Chattopadhyay, A. Reducing depth of quantum adder using ling structure. In Proceedings of the 2023 IFIP/IEEE 31st International Conference on Very Large Scale Integration (VLSI-SoC), Dubai, United Arab Emirates, 16–18 October 2023; pp. 1–6.
21. Gossett, P. Quantum carry-save arithmetic. *arXiv* **1998**, arXiv:quant-ph/9808061.
22. Takahashi, Y.; Kunihiro, N. A fast quantum circuit for addition with few qubits. *Quantum Inf. Comput.* **2008**, *8*, 636–649. [CrossRef]

23. Takahashi, Y.; Tani, S.; Kunihiro, N. Quantum addition circuits and unbounded fan-out. *arXiv* **2009**, arXiv:0910.2530. [CrossRef]
24. Weinstein, Y.S.; Pravia, M.; Fortunato, E.; Lloyd, S.; Cory, D.G. Implementation of the quantum Fourier transform. *Phys. Rev. Lett.* **2001**, *86*, 1889. [CrossRef]
25. Coppersmith, D. An approximate Fourier transform useful in quantum factoring. *arXiv* **2002**, arXiv:quant-ph/0201067.
26. Draper, T.G. Addition on a quantum computer. *arXiv* **2000**, arXiv:quant-ph/0008033.
27. Cheng, K.W.; Tseng, C.C. Quantum full adder and subtractor. *Electron. Lett.* **2002**, *38*, 1343–1344. [CrossRef]
28. Ruiz-Perez, L.; Garcia-Escartin, J.C. Quantum arithmetic with the quantum Fourier transform. *Quantum Inf. Process.* **2017**, *16*, 1–14. [CrossRef]
29. Orts, F.; Ortega, G.; Combarro, E.F.; Garzón, E.M. A review on reversible quantum adders. *J. Netw. Comput. Appl.* **2020**, *170*, 102810. [CrossRef]
30. Rattew, A.G.; Sun, Y.; Minssen, P.; Pistoia, M. The efficient preparation of normal distributions in quantum registers. *Quantum* **2021**, *5*, 609. [CrossRef]
31. Horowitz, P.; Hill, W.; Robinson, I. *The Art of Electronics*; Cambridge University Press: Cambridge, UK, 1989; Volume 2.
32. Burks, A.W.; Goldstine, H.H.; Von Neumann, J. Preliminary discussion of the logical design of an electronic computing instrument. In *The Origins of Digital Computers: Selected Papers*; Springer: Berlin/Heidelberg, Germany, 1946; pp. 399–413.
33. Lim, Y.L.; Beige, A.; Kwek, L.C. Repeat-until-success linear optics distributed quantum computing. *Phys. Rev. Lett.* **2005**, *95*, 030505. [CrossRef]
34. Wiebe, N.; Roetteler, M. Quantum arithmetic and numerical analysis using Repeat-Until-Success circuits. *arXiv* **2014**, arXiv:1406.2040. [CrossRef]
35. Bocharov, A.; Roetteler, M.; Svore, K.M. Efficient synthesis of universal repeat-until-success quantum circuits. *Phys. Rev. Lett.* **2015**, *114*, 080502. [CrossRef] [PubMed]
36. Li, J. Elementary Quantum Arithmetic Logic Units for Near-Term Quantum Computers. *arXiv* **2024**, arXiv:2408.06561.