Finite Memory Quantum Computer and Synchronous Quantum Circuits

Arinta Auza, supervised by Emmanuel Jeandel and Simon Perdrix
CARTE Team, LORIA-INRIA Nancy-Grand Est

The general context

Quantum computation uses quantum mechanical phenomena in its implementation as opposed to Newtonian mechanics for classical computation. There are several ways to do computation in quantum computer including the now well-studied quantum circuit model. In the future, it is expected to have a large scale quantum computer in which we will only presented with some finite number of qubits to perform the computation. Hence, we should consider a model where we only have some finite register for the qubits. One question that we need to ask is which problems are implementable in this model.

The research problem

My internship focuses on some implementation on the finite memory quantum circuits model. There are several studies on quantum circuits, however there is still not study on the finite memory model so far. In order to understand the model, we start by considering the classical counterpart. The next step is we look at reversible circuits as we know that quantum circuits are reversible. We find that there is an example of problem that is not implementable on the reversible circuits with finite memory. We also prove that this problem is not implementable in the quantum model. After that, we study several implementable examples.

Your contribution

My contribution on this internship is to present several implementable problems and how to implement these problems on a quantum circuit with finite memory. There are some properties that we need to check before we can be sure that certain problem is implementable.

Arguments supporting its validity

In order to support the validity on the circuits presented on my results, I give some proofs that certain kind of outputs is needed in order to preserve the orthogonality that is required by quantum circuits which have to compute unitary functions. Then after the circuits are presented, I check if they give the expected results.

Summary and future work

In this internship, we limit the scope of the study to only consider some finite classical bits as the inputs to the circuits. We might want to see what will happen if we use any inputs and give some description of the evolution. We can also ask what we can do for several problems that are not implementable, such as Addition that is proved in this report to be unimplementable. Another interesting question is how to give formalization on the synchronous quantum circuits model. There is a study on how to formalize synchronous classical circuits using 2-adic numbers. However, this will need a more complicated mathematical tools to answer the question.

1 Introduction

The use of quantum mechanical phenomena in computer science opens huge prospects, both algorithmic (such as Shor's algorithm factorization in polynomial time [Sho94]) and cryptography (such as quantum key exchange BB84 [BB84] which has an unconditional security compared to classical public key cryptographic protocol which relies on the computational hardness assumption).

Concerning the implementation of a quantum computer, the current prototypes have a memory of a few dozen quantum bits while several thousand are needed to implement the factorization algorithm on instances big enough to have a significant acceleration against classical computer. Given this situation, the aim of this internship is to study the quantum synchronous circuits with a finite memory. The main objective is to characterize the unitary transformations that can be implemented by such synchronous circuits, that is to say which can be implemented with a finite memory.

We will introduce classical synchronous circuit in Section 2 and give formal definition for this model. Then we will present finite memory reversible circuit in Section 3 as many quantum circuits are modeled from reversible circuits. We will see that there is some problem that cannot be implemented in finite memory reversible circuits, as in this report we consider Addition problem. The next is to define the quantum synchronous circuits using finite memory in Section 4. We will also proof that the same problem is not implementable in this model. Hence, there is also limitation on using this model. In Section 5 we will see several examples that are implementable in finite memory quantum circuits. We use graph states as the expected outputs.

2 Classical Synchronous Circuit

A synchronous circuit is a digital circuit in which the changes in the state of memory elements are synchronized by a clock signal. This circuit operates upon infinite binary sequences where each variable x takes on consecutive binary values $x_0, x_1, \ldots, x_t, \ldots \in \mathbb{B}$ as digital time progresses through the natural numbers $t \in \mathbb{N}$. Synchronous circuits map infinite binary sequences, representing the successive input values at each clock tick $t \in \mathbb{N}$, into infinite binary sequences, representing the corresponding output values.

Synchronous circuits consist of two main parts, i.e. the set of classical logical gates and memory to register the values that will be necessary for the computation at the next clock tick. In practice, we can only have some finite number $k \in \mathbb{N}$ of bits that can be stored in the memory where the initial values of the memory are set to zero. A tool to analyse synchronous circuits has been developped using 2-adic numbers [Vui94].

Definition 2.1 *A function $F : (\mathbb{B}^n)^{\mathbb{N}} \to (\mathbb{B}^r)^{\mathbb{N}}$ is* ***implementable*** *in a synchronous circuit $f : \mathbb{B}^{n+k} \to \mathbb{B}^{r+k}$ if it exists which satisfies*

$$F(x) = y \textit{ iff } \exists m \in \mathbb{B}^k \textit{ s.t. } f(x_t, m_t) = (y_t, m_{t+1}) \textit{ and } m_0 = 0$$

where $n, r, k \in \mathbb{N}$.

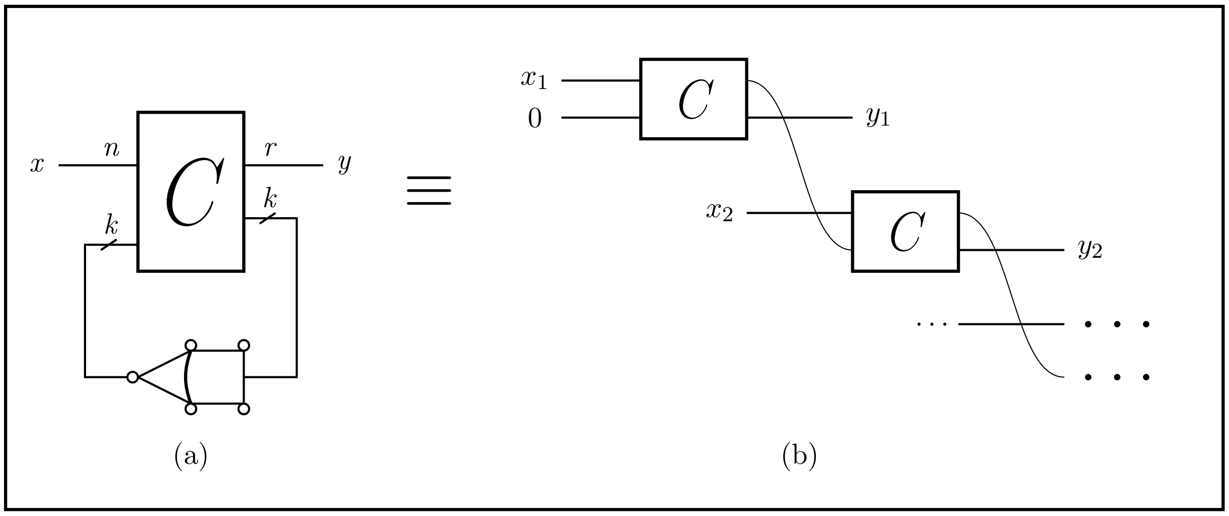


Figure 1: Circuits with finite memory. (a) The overall picture of the circuit (b) Unfolding the circuit

The implementation of a circuit f that computes F defined above can be seen in Figure 1(a). We can unfold the circuit as in Figure 1(b) to see the operation in each clock tick t. We have x_t, m_t as the values of the input and memory respectively at clock tick t and y_t as the corresponding output.

Example 2.2 *In order to see how this circuit works we consider addition problem which is defined as follow :*

Input : $a, b \in \mathbb{N}$

Output : $s = a + b$

Suppose

$$a = a_0 + 2 \cdot a_1 + 2^2 \cdot a_2 + \cdots + 2^u \cdot a_u + 2^{u+1}.0 + 2^{u+2} \cdot 0 + \cdots$$
$$b = b_0 + 2 \cdot b_1 + 2^2 \cdot b_2 + \cdots + 2^v \cdot b_v + 2^{v+1}.0 + 2^{v+2} \cdot 0 + \cdots$$

We do the computation bit-wise starting from (a_0, b_0) to the infinity. The circuit for this problem is given by Figure 2 [Ber13].

The circuit is built by set of gates in Figure 3 and requires only one memory. At each clock tick t the circuit computes

$$s_t = (a_t + b_t + m_t)(\text{mod } 2)$$
$$m_{t+1} = (a_t + b_t + m_t)/2$$

where $m_0 = 0$ and m_{t+1} as the carry is stored in the memory for the computation at clock tick $t + 1$. As a result, we have

$$a_t + b_t + m_t = s_t + 2m_{t+1}$$

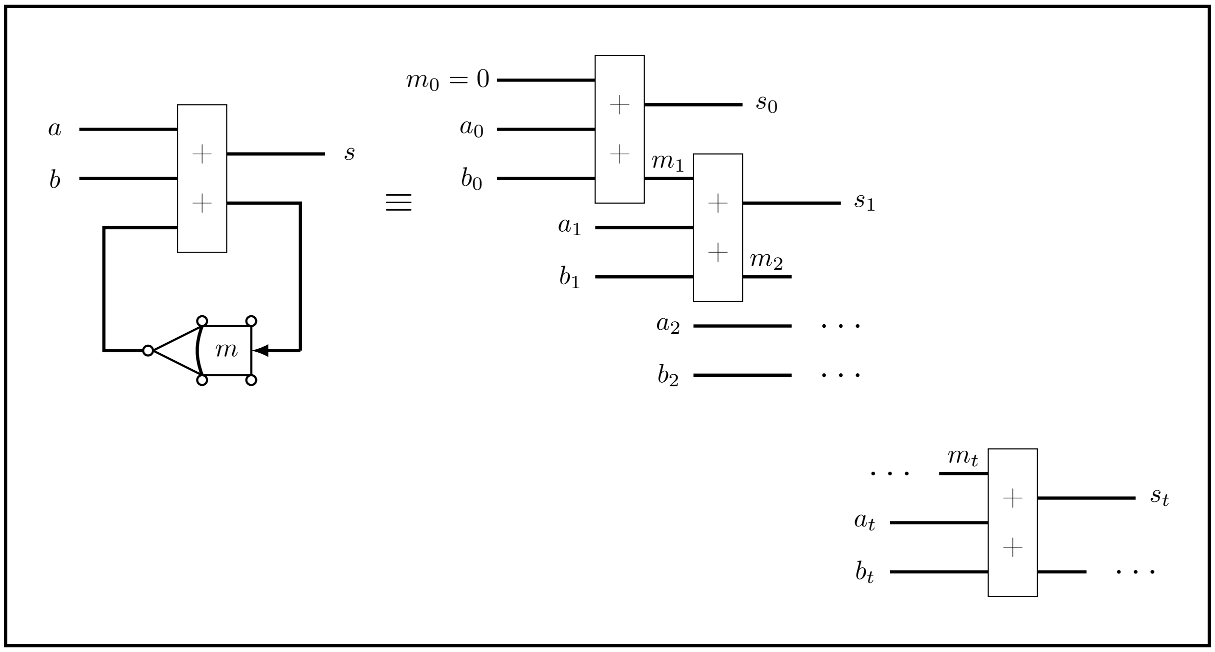


Figure 2: Circuit for addition

3 Finite Memory Reversible Circuit

The classical gates used to perform computation are typically irreversible and in most cases we have less output bits than the input bits. This loss of information can cause energy loss. According to Landauer a dissipation of the order kT, where k is the Boltzmann constant and T is the temperature of the circuit, is needed for each irreversible function[Lan61].

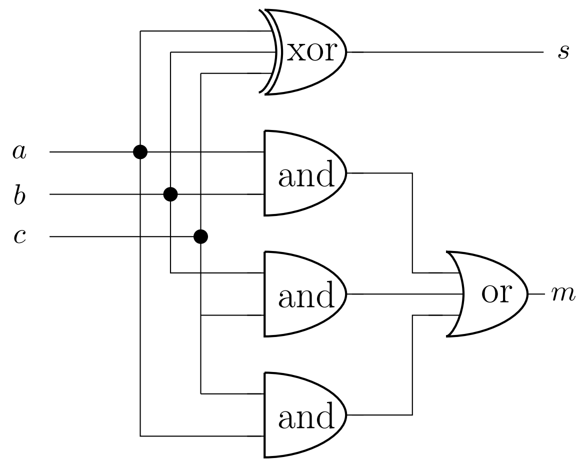
However Charles Bennett found that any computation can be performed by reversible steps which in principle requires no dissipation and no power expenditure[Ben73]. Reversible computation can run both forward and backward, hence we can recover inputs from outputs and can stop and go back to any point in the computation. Landauer [Lan61]introduced reversible circuits to perform reversible computation and later formalized by Toffoli and Fredkin [Tof80; FT82].

Definition 3.1 *A gate is **reversible** if the (Boolean) function it computes is bijective [She+03].*

In order to ensure the bijectivity, the gates must have the same number of input wires and output wires. A gate with k inputs and k outputs is called a $k * k$ gate[Per+01].

Now we will introduce a concept where the reversible circuits have memory that can only store some finite number of bits. There are differences on the functions that this circuit can compute compared to the reversible circuits that we already know as we will see in several examples.

Definition 3.2 *A circuit $f : \mathbb{B}^{n+k} \to \mathbb{B}^{n+k}$ is called **a finite memory reversible circuit** if it is constructed by reversible gates and has k bits memory for some finite number $k \in \mathbb{N}$.*



$s = a$ xor b xor c
$m = (a$ and $b)$ or $(b$ and $c)$ or $(c$ and $a)$

Figure 3: Gates for addition

Definition 3.3 *A function $F : (\mathbb{B}^n)^{\mathbb{N}} \to (\mathbb{B}^n)^{\mathbb{N}}$ has a* ***finite memory reversible circuit implementation*** *f, iff there exists a circuit f finite and reversible that computes F.*

Although the circuit f consists of set of gates that compute bijective function, the function F is not necessarily bijective. We will show this in Example 3.4. In addition, there is also some cases where unique inputs cannot be recovered from the outputs even though the function F that it computes is bijective as we can see in Example 3.5. Hence, unlike reversible circuits that we already know there are several cases where we cannot recover unique input from the output.

Example 3.4 *The finite memory reversible circuit in Figure 4 computes a non-invertible function F.*

In order to see that F is non-invertible, we take two different inputs 010^{ω} and 110^{ω}. On these two inputs, the circuit will compute the same output 0010^{ω}. Hence, F is not one-to-one. Although the evolution is reversible because the gates being used are reversible, we cannot recover the input bits only from the output without considering what is on the memory. At the end of the computation for these particular inputs, the memory will store different bits. The value stored in the memory is 0 on input 010^{ω}, whereas we have 1 on input 110^{ω}.

Example 3.5 *The circuit in Figure 5 is reversible with finite memory and computes a bijective function F. However, there is no finite memory reversible implementation for F^{-1}.*

This circuit computes

$$F : x_0 x_1 \cdots \mapsto y_0 y_1 \cdots \text{ where } y_0 = x_0 \text{ and } y_i = x_{i-1} \oplus x_i \tag{1}$$

We will see in Lemma 3.6 that the inverse of this function which computes

$$F^{-1} : y_0 y_1 \cdots \mapsto x_0 x_1 \cdots \text{ where } x_i = \sum_{j=0}^{i} y_j \tag{2}$$

does not have finite memory reversible implementation.

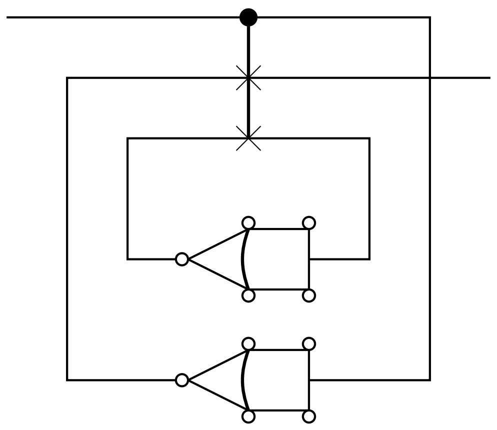


Figure 4: Non bijective function that is reversibly implementable

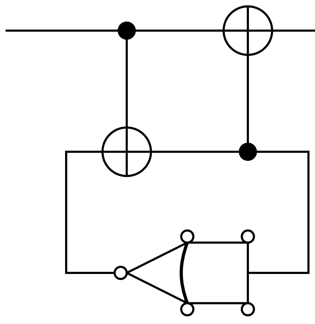


Figure 5: A function F with no finite memory reversible implementation for F^{-1}

Lemma 3.6 *The inverse of function F in Equation 1 that is given by Equation 2 is not implementable in reversible circuit with finite memory.*

Proof. We will prove this by contradiction. Suppose there exists finite memory reversible circuit f that computes F^{-1}. We define

$$C_p = \{m \in \mathbb{B}^k \mid \exists m > 0, \exists x, y \in \mathbb{B}^n \text{ s.t } \sum_{i=1}^{n} x_i = p \bmod 2 \wedge f^{(n)}(0^k, x) = (y, m)\}$$

where $\forall j \leq n, y_j = \sum_{i=0}^{j} x_i$ and $f^{(n)}$ is inductively defined as $f^{(0)} = I_k$ and $f^{(n)} = (I_1 \times f^{(n-1)}) \circ (f_1 \times I^{(n-1)})$.

Let $G_p : C_p \to C_p$ where for all $m \in C_p$ we have $f(m, 0) = (p, G_p(m))$. Since f is bijective, G_p must be injective hence bijective. Therefore, G_1 is bijective. Let us consider $f(0^k, 1) = (1, m_0)$ for some $m_0 \in \mathbb{B}^k$. We have $f(G_1^{-1}(m_0), 0) = (1, m_0) = f(0^k, 1)$ which contradicts that f is bijective.

□

3.1 Addition

In Example 2.2 we have seen the implementation of addition of two natural numbers in classical synchronous circuit. We will need a new definition for the reversible implementation of this problem as the same number of input and output bits is required.

Definition 3.7 *Let us define* ***Addition*** *as a function $F : \mathbb{B}^{\mathbb{N}} \to \mathbb{B}^{\mathbb{N}}$ where for all $a, b \in \mathbb{B}$ we have $F(a, b) = (a, a + b)$.*

3.1.1 No Go Result for Addition

In this section, we will study the implementation of addition in reversible circuit with finite memory. The objective is to help the construction for the quantum implementation as quantum evolution must be reversible [NC00]. However, as we can see in the following lemma, this problem is not implementable in reversible circuit with finite memory.

Lemma 3.8 *There is no finite memory reversible synchronous circuit that implements f defined in Definition 3.7.*

Proof. By contradiction, assume there exists $j \geq 0$ and finite memory reversible circuit $f : \mathbb{B}^{k+2} \to \mathbb{B}^{k+2}$ that implements F. Let C be the set of reachable memory states that encode that there is a carry.

$$C = \{m \in \mathbb{B}^k \mid \exists n > 0, \exists a, b, s \in \mathbb{B}^n \text{ s.t. } a + b \geq 2^n \wedge f^{(n)}(0^k, a, b) = (a, s, m)\}$$

where $f^{(n)}$ is inductively defined as $f^{(0)} = I_k$ and $f^{(n)} = (I_1 \times f^{(n-1)}) \circ (f_1 \times I^{(n-1)})$.

Let $G : C \to C$ such that for any $m \in C, f(m, 1, 0) = (1, 0, G(m))$. As f is injective, G is injective, so bijective. Let $m_0 \in C$ such that $f(0^k, 1, 1) = (1, 0, m_0)$. Notice that $f(G^{-1}(m_0), 1, 0) = (1, 0, m_0) = f(0^k, 1, 1)$ which contradicts that f is reversible. □

4 Quantum Synchronous Circuit

Quantum synchronous circuits are composed of a finite number of quantum logic gates (e.g. Hadamard CNOT) and memories. Each memory can store a qubit between two clock cycles. The input of the circuit is (semi-) infinite: in each clock cycle, several qubits are injected into the circuit. Similarly, at each clock cycle several qubits are produced by the circuit.

4.1 Several Notions and Notations

Before we proceed with quantum synchronous circuits with finite memory and what we can do with this model, we will introduce several notions and notations that are used in quantum computation. The notations can be found in Table 1.

4.1.1 Qubit

In classical computation, information is stored in a unit which we call bit that can be either 0 or 1. We use different information unit in quantum computation which is qubit that can store 0, 1 and superposition of both. Consider two basis states $|0\rangle$ and $|1\rangle$ where $|\cdot\rangle$ is called "ket", a Dirac notation to represent quantum states. A single qubit can be in any superposition in the vector space $\mathbb{C}^{\{0,1\}}$.

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle \text{ s.t } |\alpha|^2 + |\beta|^2 = 1 \text{ where } |\varphi\rangle \in \mathbb{C}^{\{0,1\}} \text{ and } \alpha, \beta \in \mathbb{C}$$

4.1.2 N-qubit register

We represent a system of n components with n bits in classical computation. In quantum computation this requires 2^n complex number. A quantum register of n qubits can be in any superposition in the vector space $\mathbb{C}^{\{0,1\}^n}$ which is a linear combination of the basis states.

$$|\varphi\rangle = \sum \alpha_x|x\rangle \text{ such that } \sum_{x=0}^{2^n-1} |\alpha_i|^2 = 1$$

4.1.3 Composition

Suppose we have two states $|\varphi\rangle$ of m qubits and $|\psi\rangle$ of n qubits. The composition of these two systems is denoted by a tensor product, $|\varphi\rangle \otimes |\psi\rangle$, where $\otimes$ is bilinear and for all $|x\rangle \in \mathbb{C}^{\{0,1\}^m}$ and for all $|y\rangle \in \mathbb{C}^{\{0,1\}^n}$ we have $|x\rangle \otimes |y\rangle = |xy\rangle$. The end result is a system of $m+n$ qubits.

Let $|\varphi\rangle = \sum_{x\in\{0,1\}^m} \alpha_x|x\rangle$ and $|\psi\rangle = \sum_{y\in\{0,1\}^n} \beta_y|y\rangle$. The composition of these two states is given by

$$|\varphi\rangle \otimes |\psi\rangle = \sum_{x\in\{0,1\}^m} \sum_{y\in\{0,1\}^n} \alpha_x\beta_y|xy\rangle$$

4.1.4 Inner Product

A function $(\cdot,\cdot) : \mathbb{C}^{\{0,1\}^n} \times \mathbb{C}^{\{0,1\}^n} \to \mathbb{C}$ is an **inner product** if it satisfies [NC00]:

1. For all $|x\rangle$ and $|y\rangle$ the orthogonal basis states of $\mathbb{C}^{\{0,1\}^n}$ we have
$$(|x\rangle, |y\rangle) = \begin{cases} 1, & \text{if } |x\rangle = |y\rangle \\ 0, & \text{otherwise} \end{cases}$$
2. $(\cdot,\cdot)$ is linear in the second argument,
$$\left(|v\rangle, \sum_i \lambda_i|w_i\rangle\right) = \sum_i \lambda_i \left(|v\rangle, |w_i\rangle\right)$$
3. $(|v\rangle, |w\rangle) = (|w\rangle, |v\rangle)^*$.
4. $|(|v\rangle, |v\rangle)| \geq 0$ with equality if and only if $|v\rangle = 0$.

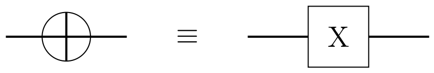
Notation	Description
α^*	Complex conjugate of α where $\alpha \in \mathbb{C}$.
$\lvert\psi\rangle$	Dirac notation of vector. We call it *ket*.
$(\lvert\varphi\rangle, \lvert\psi\rangle)$	Inner product between vector $\lvert\varphi\rangle$ and $\lvert\psi\rangle$
$\lvert\varphi\rangle \otimes \lvert\psi\rangle$	Tensor product of $\lvert\varphi\rangle$ and $\lvert\psi\rangle$
$A^\dagger$	Hermitian conjugate of operator A
$\lvert\varepsilon\rangle$	The state of 0-qubit register
$\lvert\psi\rangle^k$	$\underbrace{\lvert\psi\rangle \otimes \lvert\psi\rangle \otimes \cdots \otimes \lvert\psi\rangle}_{k \text{ times}}$
$\lvert\psi\rangle^*$	$\underbrace{\lvert\psi\rangle \otimes \lvert\psi\rangle \otimes \cdots \otimes \lvert\psi\rangle}_{k \geq 0 \text{ times}}$
$\lvert\psi\rangle^+$	$\underbrace{\lvert\psi\rangle \otimes \lvert\psi\rangle \otimes \cdots \otimes \lvert\psi\rangle}_{k > 0 \text{ times}}$
$\lvert\psi\rangle^\omega$	$\lvert\psi\rangle \otimes \lvert\psi\rangle \otimes \cdots$

Table 1: Several notations.

4.2 Quantum Logic Gates

A quantum logic gate is a basic quantum circuit operating on a small number of qubits. In the classical synchronous circuits we have classical logic gates as the basic building blocks, whereas we have quantum gates for quantum circuits. Unlike many classical logic gates, quantum logic gates are reversible. The computation performed by quantum gates must be *unitary*, i.e. preserve the inner product. We will introduce quantum gates that will be used throughout the report.

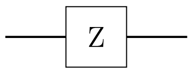
4.2.1 Not Gate.



The Not gate acts on a single qubit. This is equivalent with the NOT gate we have in classical gate. The gate computes

$$\begin{aligned} X : &\lvert 0\rangle \mapsto \lvert 1\rangle \\ &\lvert 1\rangle \mapsto \lvert 0\rangle \end{aligned}$$

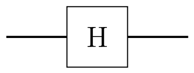
4.2.2 Z Gate.



The Z gate acts on a single qubit. It leaves the basis state $\lvert 0\rangle$ unchanged and maps $\lvert 1\rangle$ to $-\lvert 1\rangle$. This gate is also called phase-flip gate.

$$\begin{aligned} Z : &\lvert 0\rangle \mapsto \lvert 0\rangle \\ &\lvert 1\rangle \mapsto -\lvert 1\rangle \end{aligned}$$

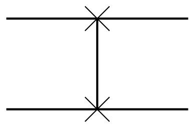
4.2.3 Hadamard gate.



The Hadamard gate operates on a single qubit. The result of the transformation by this gate will have the same probability to be measured as $|0\rangle$ or $|1\rangle$.

$$\begin{aligned} H : & |0\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ & |1\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned}$$

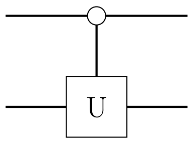
4.2.4 Swap gate.



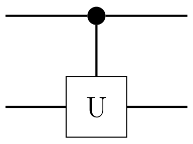
The Swap gate acts on two qubits. This gate will swap the position of the two input qubits as we can see below

$$S : |\alpha\rangle|\beta\rangle \mapsto |\beta\rangle|\alpha\rangle \text{ where } \alpha, \beta \in \mathbb{B}$$

4.2.5 Controlled gate.



$C_0 - U$

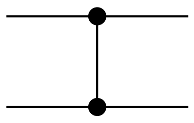


$C_1 - U$

The controlled gate acts on $n+1$ qubits where 1 qubit is the control qubit and the other n qubits are the qubits that will be computed by a unitary U. The unitary U is a quantum gate or a set of quantum gates that acts on n qubits. We have two different kind of controlled gate : C_0 where the control qubit is $|0\rangle$ and C_1 where the control qubit is $|1\rangle$.

$$\begin{aligned} C_0 - U : & |0\rangle|\psi\rangle \mapsto |0\rangle \otimes U|\psi\rangle \\ & |1\rangle|\psi\rangle \mapsto |1\rangle \otimes |\psi\rangle \\ C_1 - U : & |0\rangle|\psi\rangle \mapsto |0\rangle \otimes U|\psi\rangle \\ & |1\rangle|\psi\rangle \mapsto |1\rangle \otimes |\psi\rangle \end{aligned}$$

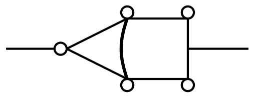
4.2.6 Controlled-Z.



We use special notation for Controlled-Z because the outcome of having the control qubit in the first qubit is equivalent with the outcome of having the control qubit in the second qubit. As we can see, the gate will change the phase only when the first and the second qubit are $|1\rangle$ and does nothing to other inputs on the basis states $|00\rangle, |01\rangle, |10\rangle$.

$$\Lambda Z : |\alpha\rangle|\beta\rangle \mapsto (-1)^{\alpha\beta}|\alpha\rangle|\beta\rangle \text{ where } \alpha, \beta \in \mathbb{B}$$

4.2.7 Memory.



The memory stores the qubits that will be used in the next computation. We will use some finite number of memory in the computation.

4.3 Finite Memory Quantum Circuit

Now we will give the definition of finite memory quantum circuit as we are only interested in circuits with finite number of qubits in the memory.

Definition 4.1 *A circuit $t : \mathbb{C}^{\{0,1\}^{n+k}} \to \mathbb{C}^{\{0,1\}^{n+k}}$ is called **a finite memory quantum circuit** if it is constructed by quantum logic gates and has k qubits memory for some finite number $k \in \mathbb{N}$.*

Definition 4.2 *A function $T : (\mathbb{C}^{\{0,1\}^n})^{\mathbb{N}} \to (\mathbb{C}^{\{0,1\}^n})^{\mathbb{N}}$ has a **finite memory quantum circuit implementation** t, iff there exists a finite memory quantum circuit t that computes T.*

Note that the function T computed by finite memory quantum circuit t is not necessarily unitary. We will see this in Example 4.3.

Example 4.3 *Let T be a function computed by a quantum circuit t that is defined in Figure 6. The function T is not unitary.*

To see this function is not unitary, let us take $\frac{1}{2}(|00\rangle + |11\rangle)$ as input. The circuit will produce a mixed state as the output is entangled with the qubits in the memory, hence T is not unitary.

Lemma 4.4 *Let t be a finite memory quantum circuit that computes T where all the values in the memory at the end of the computation is 0 for all finite input $|x\rangle$. Then T is unitary.*

Proof. Let $|x\rangle$ and $|y\rangle$ be any finite inputs in $\mathbb{C}^{\{0,1\}^n}$. Suppose $t^{(n)}$ is the transformation on the circuit t after n qubits. We have $t^{(n)}(|x\rangle \otimes |0\rangle^k) = T(|x\rangle) \otimes |0\rangle^k$ and $t^{(n)}(|y\rangle \otimes |0\rangle^k) = T(|y\rangle) \otimes |0\rangle^k$ for some $k \in \mathbb{N}$. Since $t^{(n)}$ is unitary we have $(|x\rangle|0\rangle^k, |y\rangle|0\rangle^k) = (T(|x\rangle) \otimes |0\rangle^k, T(|y\rangle) \otimes |0\rangle^k)$. This implies $(|x\rangle, |y\rangle) = (T(|x\rangle), T(|y\rangle))$. Therefore, T is unitary. □

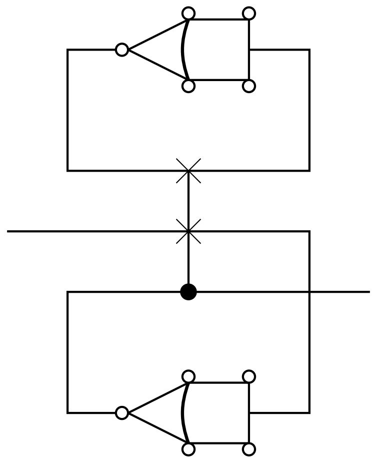


Figure 6: Non unitary function

Corollary 4.5 *For any two orthogonal finite inputs $|x\rangle$ and $|y\rangle$ on a circuit t in Lemma 4.4, we have $T(|x\rangle)$ and $T(|y\rangle)$ orthogonal.*

Proof. From Lemma 4.4, we have T is unitary. Hence $(T(|x\rangle), T(|y\rangle)) = (|x\rangle, |y\rangle) = 0$. □

4.4 Addition in Finite Memory Quantum Circuit

We have seen that there is no finite memory reversible circuit for addition in the previous chapter. Interestingly, we can prove that there is no finite memory quantum circuit for this problem as what will be shown in the following lemma.

Lemma 4.6 *There is no finite memory quantum circuit t that implements T defined in Definition 3.7.*

Proof. Assume that there exists such circuit. Let S be the set of basis states that indicate there exists a carry.

$$S = \{|\varphi\rangle \in \mathbb{C}^{\{0,1\}^k} \mid \exists n > 0, \exists a, b, s \in \mathbb{B}^n \text{ s.t. } a + b >= 2^n \wedge t^{(n)}(|0\rangle^k |a\rangle|b\rangle) = |a\rangle|b\rangle|\varphi\rangle\}$$

where $t^{(n)}$ is inductively defined as $t^{(0)} = I_k$ and $t^{(n)} = (I_1 \times t^{(n-1)}) \circ (t_1 \times I^{(n-1)})$ and $|\varphi\rangle$ is basis state of $\mathbb{C}^{\{0,1\}^k}$. We have $C = span(S)$ as the set of all reachable memory states that encode that there is a carry.

Let $F : S \to S$ such that for any $|\varphi\rangle \in S$, $t(|\varphi\rangle \otimes |10\rangle) = |10\rangle \otimes F(|\varphi\rangle)$. Since t is unitary then t is injective, so F is injective hence bijective.

Let $|\psi\rangle \in C$. Then $|\psi\rangle = \sum a_i|\varphi_i\rangle$ for $|\varphi_i\rangle \in S$ and $a_i \in \mathbb{C}$. We define $G : C \to C$ as

$G(|\psi\rangle) = \sum a_i F(|\varphi_i\rangle)$.

$$\begin{aligned} t(|\psi\rangle \otimes |10\rangle) &= t\left(\sum a_i |\varphi_i\rangle \otimes |1,0\rangle\right) \\ &= \sum a_i t(|\varphi_i\rangle \otimes |1,0\rangle) \\ &= \sum a_i \left(|10\rangle \otimes F(|\varphi_i\rangle)\right) \\ &= \sum |10\rangle \otimes (a_i F(|\varphi_i\rangle)) \\ &= |10\rangle \otimes \sum a_i F(|\varphi_i\rangle) \\ &= |10\rangle \otimes G(|\psi\rangle) \end{aligned}$$

Then G is injective. Let $|\psi_0\rangle \in C$ such that $t(|0^k\rangle|11\rangle) = |10\rangle \otimes |\psi_0\rangle$. Notice that $t(G^{-1}|(\psi_0)\rangle \otimes |10\rangle) = |10\rangle \otimes |\psi_0\rangle$. This contradicts the fact that t is unitary. □

5 Graph States

A graph state is multi-particle entangled states that correspond to mathematical graphs [MB04]. Each qubit is represented by a vertex of the graph, and there is an edge between every interacting pair of qubits. In particular, graph states is a convenient way of representing certain types of entangled states.

Definition 5.1 *Given a graph $G = (V, E)$ with the set of vertices V and the set of edges E. The corresponding graph state $|G\rangle$ is the pure state with state vector [Hei+06]:*

$$|G\rangle = \left[\prod_{(i,j)\in E} \Lambda \mathrm{Z}_{i,j}\right] |+\rangle^{\otimes V}$$

where the operator $\Lambda\mathrm{Z}_{i,j}$ is the controlled-Z interaction between the two vertices (qubits) v_i, v_j and

$$|+\rangle = \frac{|0\rangle + |1\rangle}{2}$$

corresponding to the empty graph.

5.1 Preparation of Graph States

For a given graph state G, we prepare each of the vertex on the corresponding graph as a qubit in the state $|+\rangle$ as we can see in Figure 7[MP05].

The next step is to apply the unitary transformation $\Lambda\mathrm{Z}_{ij}$ on the qubits v_i and v_j for each edge $v_i v_j$ on the graph G.

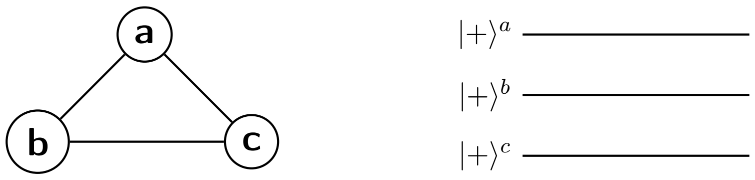


Figure 7: Graph state

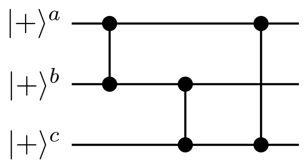


Figure 8: Graph state preparation

5.2 Several Graph States on Finite Memory Quantum Circuits

In this section, we will investigate several graph states to see if there is some finite memory quantum circuit implementation. We limit the scope of study to only consider classical bits as inputs and pure states as outputs. In order to produce such outputs we assume that for any finite input x we have $|0\rangle$ in the memory. We first examine if it is possible to produce certain graph states. We do this by verifying if any two outputs given by two orthogonal inputs are orthogonal according to Corollary 4.5. The next step is to give the complete circuit for the corresponding graph state if they exist.

We define the problem as follows :

Input : $|0\rangle^*(|1\rangle^{k_i}|0\rangle^+)^*|0\rangle^\omega$

Expected Output : $|0\rangle^*(|G_{m_i}\rangle 0^*)^*|0\rangle^\omega$

where G_{m_i} is the graph state of m_i vertices that we want to produce and $m_i = \nu(k_i)$. In the cases that we will study in the next section we have $m_i = k_i + j$ where $j \in \{0, 1\}$.

5.2.1 Path

Definition 5.2 *A **path** P_n is a graph (V, E) of n vertices where $V = \{v_1, v_2, \cdots, v_n\}$ and for all $1 \leq i \leq n-1$ we have $v_i v_{i+1} \in E$.*

5.2.1.1 Orthogonality

Before we present the circuit that produce such graph state, we need to examine the orthogonality of the expected output. Suppose we have two inputs $|0^\omega\rangle$ and $|10^\omega\rangle$. These two inputs are orthogonal and by Corollary 4.5, we must produce orthogonal outputs. However since the expected outputs are $|0^\omega\rangle$ and $|P_1^\omega 0\rangle$, the inner product of these two states is

$$\begin{aligned}(|0^\omega\rangle, |P_1 0^\omega\rangle) &= (|0\rangle, |+\rangle)\\ &\neq 0\end{aligned}$$

This implies that we cannot produce a circuit for this kind of output. One possibility is to put a one qubit state before and/or after the produced graph state, i.e. we expect $|0\rangle^*(|\varphi\rangle P_{m_i}|\psi\rangle 0^*)^*|0\rangle^\omega$ as the output.

Case 1. $\psi = \varepsilon$. We can consider $|O_1\rangle = |\varphi\rangle|P_1\rangle|0\rangle|0\rangle^\omega$ and $|O_2\rangle = |\varphi\rangle|P_2\rangle|0\rangle^\omega$ as counter examples. The inner product of these two states are given by

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (|+\rangle|0\rangle, \Lambda\mathrm{Z}|++\rangle)\\ &= (\Lambda\mathrm{Z}|+\rangle|0\rangle, |++\rangle)\\ &= (|+\rangle|0\rangle, |++\rangle)\\ &= (|0\rangle, |+\rangle)\\ &\neq 0\end{aligned}$$

Case 2. $|\varphi\rangle = |\varepsilon\rangle$. Similar to Case 1, we have $|O_1\rangle = |0\rangle|P_1\rangle|\psi\rangle|0\rangle^\omega$ and $|O_2\rangle = |P_2\rangle|\psi\rangle|0\rangle^\omega$ as counter examples.

Case 3. $|\varphi\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$ and $|\psi\rangle = \alpha_2|0\rangle + \beta_2|1\rangle$. Consider two outputs $|O_1\rangle = |\varphi\rangle|P_2\rangle|\psi\rangle|0\rangle^\omega$ and $|O_2\rangle = |\varphi\rangle|P_1\rangle|\psi\rangle|0\rangle^\omega$. The inner product of these two vectors is

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (\Lambda\mathrm{Z}|++\rangle \otimes |\psi\rangle, |+\rangle|\psi\rangle|0\rangle)\\ &= (|++\rangle|\psi\rangle, \Lambda\mathrm{Z}|+\rangle|\psi\rangle \otimes |0\rangle)\\ &= \alpha_2\,(|++\rangle|\psi\rangle, \Lambda\mathrm{Z}|+\rangle|0\rangle \otimes |0\rangle) + \beta_2\,(|++\rangle|\psi\rangle, \Lambda\mathrm{Z}|+\rangle|1\rangle \otimes |0\rangle)\\ &= \alpha_2\,(|++\rangle|\psi\rangle, |+\rangle|00\rangle) + \beta_2\,(|++\rangle|\psi\rangle, |-\rangle|10\rangle)\\ &= \alpha_2\,(|+\rangle|\psi\rangle, |00\rangle)\\ &= \frac{1}{\sqrt{2}}\alpha_2(|\psi\rangle, |0\rangle)\end{aligned}$$

If we want the inner product to be equal to zero we must have $|\psi\rangle = |1\rangle$. Analogously, we must have $|\varphi\rangle = |1\rangle$ on outputs $|O_3\rangle = |\varphi\rangle|P_2\rangle|\psi\rangle|0\rangle^\omega$ and $|O_4\rangle = |0\rangle|\varphi\rangle|P_1\rangle|\psi\rangle|0\rangle^\omega$. In this case, the only possible configurations which potentially give non-zero inner product are $|O_5\rangle = |1\rangle|P_k\rangle|1\rangle|0\rangle^\omega$ and $|O_6\rangle = (|1\rangle|P_{k_i}\rangle|1\rangle|0\rangle^{m_i})^n|0\rangle^\omega$ where $\sum_{i=1}^{n-1}(k_i + m_i) + k_n + 2(n-1) = k$.

Consider the inner product of the (k_1+2) first qubits from the right. We have

$$\begin{aligned}\left(|1\rangle \otimes \left[\prod_{i=1}^{k_1} \Lambda\mathrm{Z}_{i,i+1}\right] |+\rangle^{k_1+1}, |1\rangle \otimes \left[\prod_{i=1}^{k_1-1} \Lambda\mathrm{Z}_{i,i+1}\right] |+\rangle^{k_1} \otimes |1\rangle\right) &= (\Lambda\mathrm{Z}|++\rangle, |+\rangle|1\rangle)\\ &= (|++\rangle, \Lambda\mathrm{Z}|+\rangle|1\rangle)\\ &= (|++\rangle, |-\rangle|1\rangle)\\ &= 0\end{aligned}$$

Therefore, this will be a good candidate to be implemented on our circuit.

5.2.1.2 Circuit

By using the candidate from Case 3 above, we build a circuit in Appendix A Figure 9 which consists of five gates and two memories. Initially we set the memories to $|0\rangle$. The description of the problem computed by this circuit is as follow :

Input : $|0\rangle^*(|1\rangle^{k_i}|00\rangle|0\rangle^*)^*|0\rangle^\omega$

Output : $|0\rangle^*(|1\rangle|P_{k_i}\rangle|1\rangle 0^*)^*|0\rangle^\omega$

We need at least two $|0\rangle$s between two sequences of $|1\rangle$s in order to make sure that we get the expected graph state as each $|0\rangle$ next to $|1\rangle$ corresponds to the marker $|1\rangle$ on the output. As an example we try to compute $|11100011\rangle|0\rangle^\omega$ to get the output $|1\rangle|P_3\rangle|1\rangle|0\rangle|1\rangle|P_2\rangle|1\rangle|0\rangle^\omega$. We can see the complete evolution of the value in the memories in Appendix A Figure 10 where Q_k denotes path with one loop at the beginning, P_k^1 denotes one qubit entangled to P_k, and Q_k^1 denotes one qubit entangled to Q_k.

5.2.2 Star

Definition 5.3 *A **star** S_n is a tree on n nodes with one node having vertex degree $n-1$ and the other $n-1$ having vertex degree 1.*

5.2.2.1 Orthogonality

As what we have done in previous problem, we examine the inner product for each values of $|\psi\rangle$ and $|\varphi\rangle$. Here we consider the case where the center of the star is on the last input 1 of each sequence. The case where the center is located on any arbitrary place gives the same result.

Case 1. $|\psi\rangle = |\varphi\rangle = |\varepsilon\rangle$. Consider $|O_1\rangle = |0\rangle^\omega$ and $|O_2\rangle = |S_2\rangle|0\rangle^\omega$ on input $|0\rangle^\omega$ and $|11\rangle|0\rangle^\omega$ respectively. If we compute the inner product between the two states we will have

$$\begin{aligned}(|O1\rangle, |O2\rangle) &= (|00\rangle, \Lambda\mathrm{Z}|++\rangle)\\ &= (\Lambda\mathrm{Z}|00\rangle, |++\rangle)\\ &= (|00\rangle, |++\rangle)\\ &= \frac{1}{2} \neq 0\end{aligned}$$

Case 2. $|\varphi\rangle = |\varepsilon\rangle$. Consider $|O_1\rangle = |0\rangle|S_2\rangle|\psi\rangle|0\rangle^\omega$ and $|O_2\rangle = |S_3\rangle|\psi\rangle|0\rangle^\omega$. The inner product of these two states are given by

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (|0\rangle \otimes \Lambda Z| + +\rangle, \Lambda Z_{1,2}| + ++\rangle) \\ &= (|0\rangle| + +\rangle, \Lambda Z_{1,3}\Lambda Z_{1,2}\Lambda Z_{2,3}| + ++\rangle) \\ &= (\Lambda Z_{1,3}\Lambda Z_{1,2}|0\rangle| + +\rangle, \Lambda Z_{2,3}| + ++\rangle) \\ &= (|0\rangle| + +\rangle, \Lambda Z_{2,3}| + ++\rangle) \\ &= (|0\rangle, |+\rangle) \cdot (| + +\rangle, \Lambda Z| + +\rangle) \\ &= \frac{1}{\sqrt{2}} \cdot \frac{1}{2} \neq 0\end{aligned}$$

Case 3. $|\psi\rangle = |\varepsilon\rangle$. Consider $|O_1\rangle = |\varphi\rangle|S_2\rangle|0\rangle^\omega$ and $|O_2\rangle = |\varphi\rangle|S_3\rangle|0\rangle^\omega$. The inner product of these two states is

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (\Lambda Z| + +\rangle \otimes |0\rangle, \Lambda Z_{1,3}\Lambda Z_{1,2}| + ++\rangle) \\ &= (| + +\rangle|0\rangle, \Lambda Z_{1,3}| + ++\rangle) \\ &= (|+\rangle|0\rangle, \Lambda Z| + +\rangle) \\ &= \frac{1}{2} \neq 0\end{aligned}$$

Case 4. $|\varphi\rangle = \alpha_1|0\rangle + \beta_1|1\rangle, |\psi\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$. Consider $|O_1\rangle = |\varphi\rangle|S_2\rangle|\psi\rangle|0\rangle^\omega$ and $|O_2\rangle = |\varphi\rangle|S_3\rangle|\psi\rangle|0\rangle^\omega$. The inner product of these two states is

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (\Lambda Z| + +\rangle \otimes |\psi\rangle|0\rangle, \Lambda Z_{1,3}\Lambda Z_{1,2}| + ++\rangle \otimes |\psi\rangle) \\ &= (|+\rangle|\psi\rangle|0\rangle, \Lambda Z| + +\rangle \otimes |\psi\rangle) \\ &= (|0\rangle, |\psi\rangle) \cdot (|+\rangle|\psi\rangle, \Lambda Z| + +\rangle) \\ &= (|0\rangle, |\psi\rangle) \cdot [\alpha_2^*(|+\rangle|0\rangle, \Lambda Z| + +\rangle) + \beta_2^*(|+\rangle|1\rangle, \Lambda Z| + +\rangle)] \\ &= \alpha_2^*(|0\rangle, |\psi\rangle) \cdot (|+\rangle, |0\rangle)\end{aligned}$$

Hence, we must have $|\psi\rangle = |1\rangle$ for these two states to be orthogonal. On outputs $|O_2\rangle$ and $|O_3\rangle = |0\rangle|\varphi\rangle|S_2\rangle|\psi\rangle|0\rangle^\omega$, we must have $|\varphi\rangle = |1\rangle$. Now we consider $|O_4\rangle = |1\rangle|S_{10}\rangle|1\rangle|0\rangle^\omega$ and $|O_5\rangle = (|1\rangle|S_2\rangle|1\rangle)^3|0\rangle^\omega$. The inner product is given by

$$\begin{aligned}(|O_4\rangle, |O_5\rangle) &= \left(\prod_{i=2}^{10} \Lambda Z_{1,i}|+\rangle^{10}, \Lambda Z| + +\rangle \otimes |11\rangle \otimes \Lambda Z| + +\rangle \otimes |11\rangle \otimes \Lambda Z| + +\rangle\right) \\ &= \left(\left[\prod_{i=2}^{9} \Lambda Z_{1,i}\Lambda Z_{4,5}\Lambda Z_{8,9}\right] |+\rangle^9, |+\rangle(|11\rangle| + +\rangle)^2\right) \\ &= \left(\left[\prod_{i=2}^{5} \Lambda Z_{1,i}\Lambda Z_{2,3}\Lambda Z_{4,5}\right] |+\rangle^5, |+\rangle^5\right) \\ &= \frac{7}{2^5} \neq 0\end{aligned}$$

Therefore, there is no possible finite memory quantum circuit that gives the expected output.

5.2.3 Star with One Loop

Definition 5.4 *We call $S'_n = (V, E)$ a* ***star with one loop*** *of n vertices if for all $1 \leq i \leq n-1$ we have $v_i v_n \in E$ and $v_1 v_1 \in E$.*

We modify the problem by producing star with one loop as we can see in Definition 5.4 as the outputs.

Input : $|0\rangle^*(|1\rangle^{k_i}|00\rangle|0\rangle^*)^*|0\rangle^\omega$
Output : $|0\rangle^*(|S'_{k_i+1}\rangle|1\rangle|0\rangle^*)^*|0\rangle^\omega$.

5.2.3.1 Orthogonality

We will prove that the modified problem satisfies the orthogonality property. The only possible configuration that potentially give non-zero inner product is $|O_1\rangle = |S'_k\rangle|1\rangle|0\rangle^\omega$ and $|O_2\rangle = |0\rangle^{m_0}(|S'_{k_i}\rangle|1\rangle|0\rangle^{m_i})^n|0\rangle^\omega$ where $\sum_{i=1}^{n-1}(k_i + m_i) + m_0 + k_n + n - 1 = k$. Let us consider the inner product of the k_n last qubits before $|0\rangle^\omega$.

$$\begin{aligned}(\prod_{i=1}^{k_n}\Lambda \mathrm{Z}_{i,k_n}|-\rangle|+\rangle^{k_{n-1}}, \prod_{i=1}^{k_n}\Lambda \mathrm{Z}_{i,k_n}|+\rangle^{k_n}) &= (|-\rangle|+\rangle^{k_{n-1}}, |+\rangle^{k_n}) \\ &= (|-\rangle, |+\rangle) \\ &= 0\end{aligned}$$

5.2.3.2 Circuit

The circuit is given in Appendix A Figure 11. We use seven quantum gates and three memories where all memories are set to $|0\rangle$ at the beginning of the computation. As an example we use $|1110011\rangle|0\rangle^\omega$ as an input that computes to $|S'_4\rangle|1\rangle|S'_3\rangle|1\rangle|0\rangle^\omega$ as the output. In order to achieve S' for any sequence of 1s we need to have at least two 0s that separates between two sequences as the first $|0\rangle$ next to the last $|1\rangle$ corresponds to the center of the star and the second $|0\rangle$ corresponds to the marker $|1\rangle$.

5.2.4 Complete Graph

Definition 5.5 *A* ***complete graph*** *is a graph in which each pair of graph vertices is connected by an edge. The complete graph with n graph vertices is denoted K_n and has $\frac{n(n-1)}{2}$ (the triangular numbers) undirected edges.*

Definition 5.6 *A* ***complete graph with one loop*** *K'_n is a graph (V, E) of n vertices where for all $1 \leq i, j \leq n$ and $i \neq j$ we have $v_i v_j \in E$ and $v_1 v_1 \in E$.*

We can always find a counter example for complete graph as what we did for star. Hence we modify the problem to be complete graphs with one loop K'_n as we can see in Definition 5.6.

Input : $|0\rangle^*(|1\rangle^{k_i}|0\rangle^+)^*|0\rangle^\omega$

Output : $|0\rangle^*|0\rangle(|K'_{k_i}\rangle|1\rangle|0\rangle^*)^*|0\rangle^\omega$.

We require $k_i > 1$ because we have non-zero inner product on outputs $|O_1\rangle = |K'_2\rangle|1\rangle|0\rangle^\omega$ and $|O_2\rangle = |0\rangle|K'_1\rangle|1\rangle|0\rangle^\omega$. The inner product of these two states is given by

$$\begin{aligned}(|O_1\rangle, |O_2\rangle) &= (\Lambda \mathrm{Z}|-\rangle|+\rangle, |0\rangle|+\rangle)\\ &= (|-\rangle|+\rangle, |0\rangle|+\rangle)\\ &= (|-\rangle, |0\rangle)\\ &= \frac{1}{\sqrt{2}} \neq 0\end{aligned}$$

5.2.4.1 Local Complementation

Let $G = (V, E)$ be a graph and $a \in V$. The local complement of G at a, denoted by $\tau_a(G)$ is obtained by complementing the subgraph of G induced by the neighborhood N_a of a and leaving the rest of the graph unchanged [Hei+06].

$$\tau_a : G \mapsto \tau_a(G) := G + N_a$$

Proposition 5.7 *By local complementation of a graph G at some vertex $a \in V$ one obtains an LC-equivalent graph state $|\tau_a(G)\rangle$:*

$$|\tau_a(G)\rangle = \sqrt{X_a^\dagger}\sqrt{Z_{N_a}}|G\rangle$$

Furthermore, two graph states $|G\rangle$ and $|G'\rangle$ are LC-equivalent iff the corresponding graphs are related by a sequence of local complementations, i.e. $G' = \tau_{a_1} \circ \ldots \circ \tau_{a_n}(G)$ for some $a_1, \ldots a_n \in V$.

Proof. The proof of this Proposition can be found in [Hei+06]. □

5.2.4.2 Circuit

The idea to construct the circuit for this problem is by obtaining a local complementation of star with one loop S'_n at the center of the star a i.e. the vertex which has degree $n-1$. Since N_a is a set of disjoint vertices, the complement will be a complete graph with one loop K'_n. We can see the full description of the circuit in Appendix A Figure 12.

The circuit consists of ten quantum gates and three memory. As an example we take as input the state $|1110011\rangle|0\rangle^\omega$. The output corresponds to this input is $|0\rangle|K'_3\rangle|10\rangle|K'_2\rangle|1\rangle|0\rangle^\omega$. Here we have $|0\rangle$ in front of the overall output as a delay.

6 Conclusion

We have seen that there is a limit to quantum circuits with finite memory as Addition is not implementable in this model. We try to produce several graph states and we show that within the scope of the problem, we cannot implement some of graph states such as Star and Complete graph. Hence, we have to modify the problem by adding one loop on one of the vertices.

Appendix A Circuits for Graph States

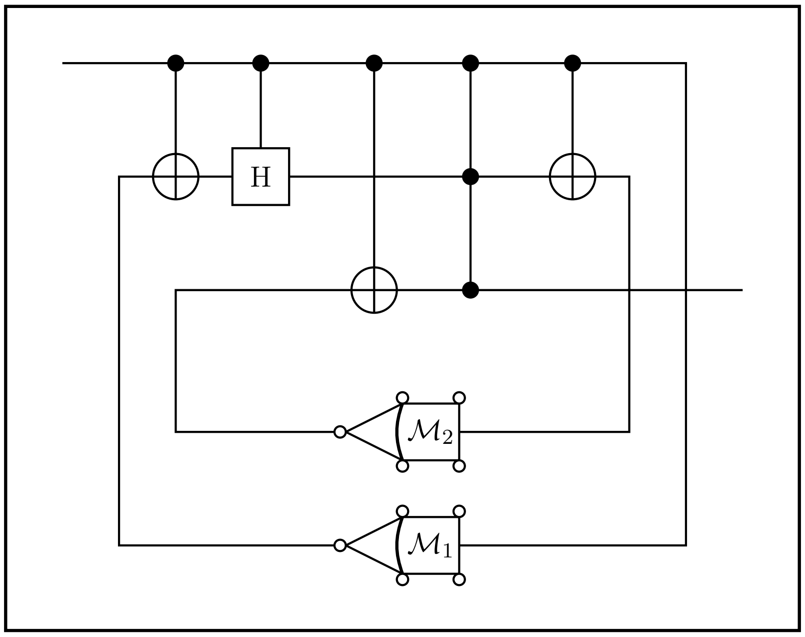


Figure 9: Circuit for path

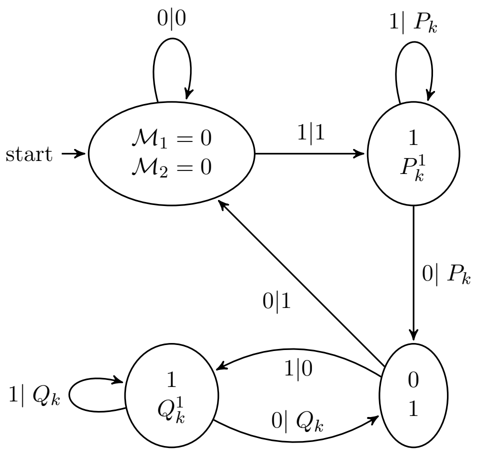


Figure 10: Evolution of memory states

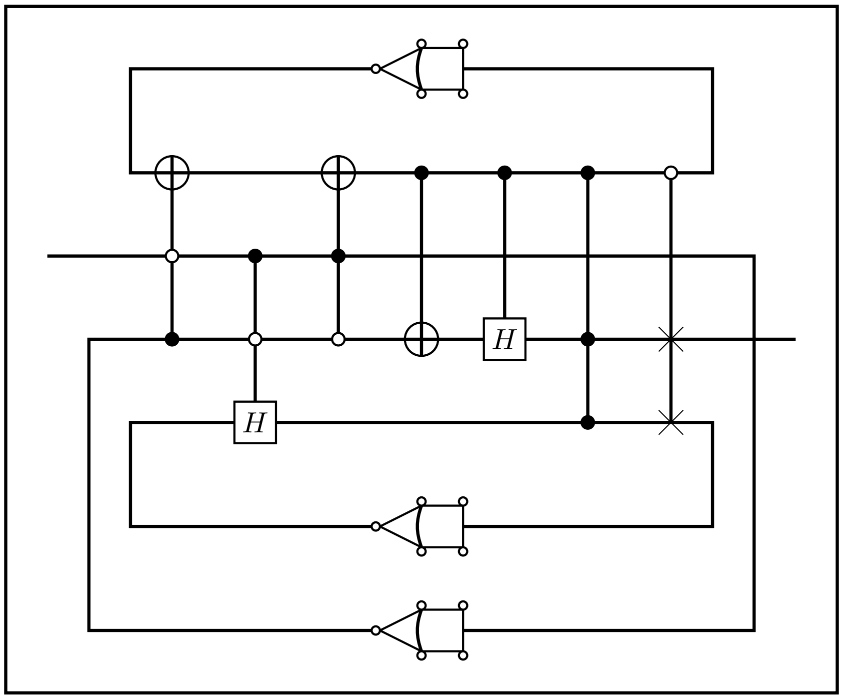


Figure 11: Circuit for star with one loop

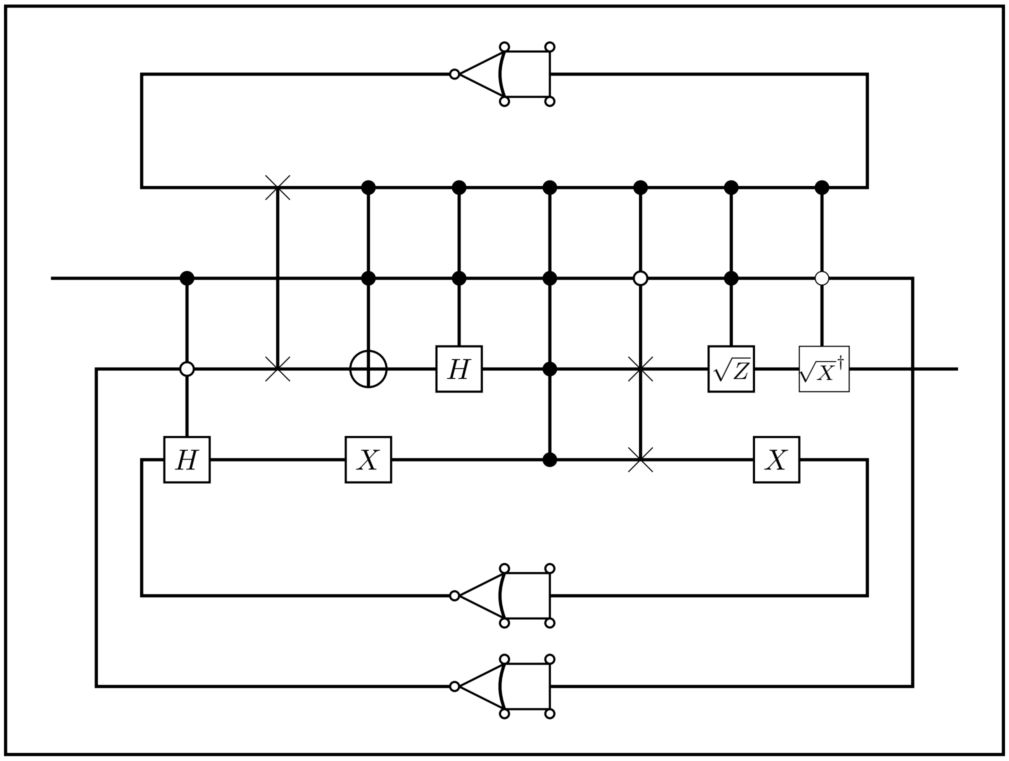


Figure 12: Circuit for complete graph with one loop

References

[BB84] C. H. Bennett and G. Brassard. “Quantum cryptography : Public key distribution and coin tossing”. In: *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India* (1984), p. 175.

[Ben73] C. H. Bennett. “Logical reversibility of computation”. In: *IBM Journal of Research and Development* 17 (1973), pp. 198–200.

[Ber13] Gerard Berry. *Playing with Time and Space in Circuits and Programs.* College de France. Chaire Algorithmes, machines et langage. 2013.

[FT82] E. Fredkin and T. Toffoli. “Conservative logic”. In: *International Journal of Theoretical Physics* 21.3/4 (1982), pp. 219–253.

[Hei+06] Marc Hein et al. “Entanglement in Graph States and its Applications”. In: *arXiv:quant-ph/0602096* (2006).

[Lan61] R. Landauer. “Irreversibility and heat generation in the computing process”. In: *IBM Journal of Research and Development* 5 (1961), pp. 183–191.

[MB04] J. Eisert M. Hein and H. J. Briegel. “Multiparty entanglement in graph states”. In: *Phys. Rev. A* 69.062311 (2004).

[MP05] Mehdi Mhalla and Simon Perdrix. *Graph States.* EPIT. 2005.

[NC00] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000. ISBN: 9780521635035.

[Per+01] M. Perkowski et al. “A General Decomposition For Reversible Logic”. In: *Reed-Muller Workshop* (2001).

[She+03] Vivek V. Shende et al. “Synthesis of Reversible Circuits”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 22.6 (2003), pp. 710–722.

[Sho94] P. Shor. “Algorithms for quantum computation : Discrete logarithms and factoring”. In: *IEEE Computer Society Press Shafi Goldwasser, editor, Proceedings of the 35nd Annual Symposium on Foundations of Computer Science* (1994), pp. 124–134.

[Tof80] T. Toffoli. “Reversible computing”. In: *Automata, Languages and Programming, 7th Colloqium* 85 (1980), pp. 632–644.

[Vui94] Jean E. Vuillemin. “On Circuit and Number”. In: *Computers, IEEE Transactions* 43.8 (1994), pp. 868–879.