中原大學資訊工程學系學士論文

**隨機性對於量子糾錯算法的重要性**

**及適用於IBM Quantum的量子糾錯算法**

作者：曾品元11027233、江庭瑄11027235、

王紫薰11027238

指導教授﹕黃琮暐教授、張元翔教授

中華民國 113 年 2 月

目錄

中文摘要

Shor's Error Correction Code

第一章 量子運算介紹

1.1 量子運算

1.2 為何需要糾錯

第二章 糾錯演算法的實作與分析

模擬器CODE

參考文獻

圖目錄

表目錄

**中文摘要**

隨著古典電腦技術的發展，它已經面臨著瓶頸。在馮紐曼架構的古典電腦體系下，資料儲存技術進展緩慢，致使古典電腦的計算能力難以實現顯著的提升。相較之下，量子電腦雖然仍屬較不成熟的技術，但在處理特定問題的時間複雜度遠優於古典電腦。

古典電腦的位元狀態僅能為1或0，而量子電腦的位元狀態可以同時為1和0。這種獨特的特性使得量子電腦的n個bit能夠同時表示2^n種狀態，使得在搜尋計算方面，量子電腦的速度遠超古典電腦。然而，正因為這種同時處於0和1的特性，量子電腦的錯誤率相對較高，糾錯變得極具挑戰性，需要採用與以往不同的思維方式來滿足糾錯需求。

**Shor's Error Correction Code**

Shor 糾錯碼（Shor's Error Correction Code）是量子糾錯碼的一種，於1995年由美國物理學家彼得·肖爾（Peter Shor）提出。其基本原理是將一個qubit的資料分佈在多個qubit上。

**重複碼**（Repetition Code）：

首先，使用重複碼將一個qubit的資料擴展到三個qubit。

對於狀態  
，其編碼狀態為  
。

**相位糾錯**：

對每一個物理qubit再編碼為三個qubit。最終編碼後的狀態是9 qubit，每三個qubit對應一個相同狀態，表示初始的一個邏輯qubit。

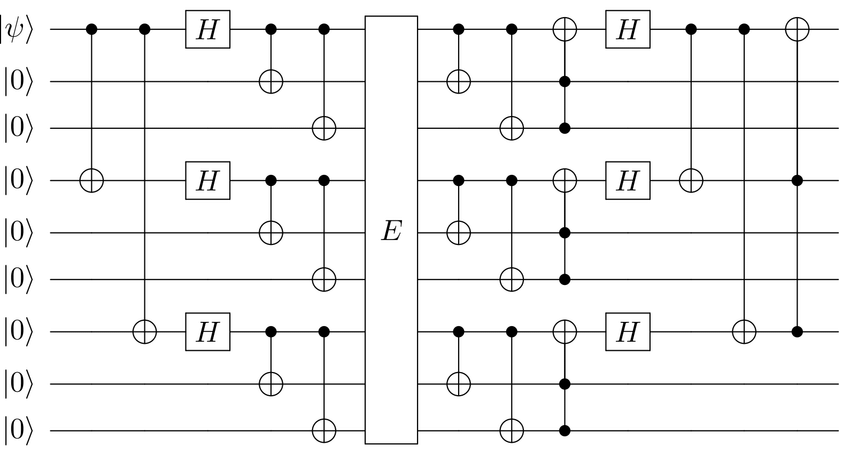
**位元翻轉錯誤**：

通過對每個三qubit組進行多數表決（majority voting），可以檢測並糾正位元翻轉錯誤。例如，如果某個三qubit組為，則判斷中間的qubit可能發生了qubit翻轉錯誤，並糾正回。

**相位翻轉錯誤**：

通過相位糾錯碼的編碼方式來檢測和糾正相位翻轉錯誤。

Shor 糾錯碼是量子計算領域的基礎技術之一，它將一個量子比特的信息分佈到多個量子qubit上，利用冗餘來檢測和糾錯量子計算過程中常見的錯誤。Shor提供了一種保護量子信息免受外界干擾和內部噪聲的方法，且能同時糾正qubit的翻轉和相位翻轉錯誤，這是單獨的重複碼或相位糾錯碼無法實現的，在理論上提供了對量子信息的強大保護，為實現實際可行的量子計算奠定了基礎。



**第一章 量子運算介紹**

**1.1量子運算**

其中表示相位，正負為方向，大小為振幅。相位的平方為機率，且和為1。

量子邏輯閘使用么正矩陣表示，而基本狀態以表示，以表示。常見的量子閘有CNOT gate、Hadamard gate、Pauli-X gate、Pauli-Y gate、Pauli-Z gate、Toffoli gate：

### CNOT gate

### 為雙量子邏輯閘,有輸入與輸出,其中輸入的位元分別稱作控制位元(control qubit)及目標位元(target qubit)。若控制位元為1則目標位元進行反閘(X閘)運算，反之則目標位元不做任何運算。類似古典位元的XOR閘。

**Hadamard gate **

是基本量子邏輯閘之一，將基本狀態變成，並將變成。

**Pauli-X gate(NOT) **

繞Bloch球體的x軸旋轉180度。

**Pauli-Y gate **

繞Bloch 球體的y軸旋轉180度。

**Pauli-Z gate A blue square with black letter z

Description automatically generated**

繞Bloch 球體的z軸旋轉180度。

### Toffoli gate (CCX gate)

### A purple dot with black lines Description automatically generated操作三個量子位元的閘,有2-qubit輸入與1-qubit輸出。若前兩個qubit皆為則對第三個qubit做Pauli-X gate運算，否則不動作。

### 用CNOT gate實作量子位元的AND閘和OR閘：

### Quantum AND gate

### A diagram of a diagram Description automatically generated

### Quantum OR Gate

### 

### 1.2為何需要糾錯

量子錯誤有很多種，以下為常見的量子錯誤:

**位元翻轉錯誤**（Bit Flip Errors）： 這種錯誤發生在量子位元（qubit）在計算過程中由於外部干擾或其他原因而由0翻轉為1，或由1翻轉為0。

隨機相位錯誤（Phase Flip Errors）： 這種錯誤涉及到量子位元的相位信息，其中量子位元可能在計算過程中受到外部因素影響，導致其相位發生錯誤。

閘操作錯誤（Gate Operation Errors）： 量子電腦使用閘操作來執行計算，這些操作可能受到外部環境的干擾，或者在實際實現中可能存在一些不完美性能，導致閘操作錯誤。

量子位元間交互錯誤（Qubit Crosstalk Errors）： 不同量子位元之間的相互作用可能導致它們之間的信息傳遞錯誤，這可能發生在相鄰的量子位元之間。

相位糾正錯誤（Phase Correction Errors）： 這種錯誤發生在嘗試糾正相位錯誤時出現的錯誤，可能導致進一步的計算錯誤。

連結錯誤（Connectivity Errors）： 由於量子位元之間的連結關係可能受到局限，連結錯誤可能會發生在計算過程中，影響計算的正確執行。

計算時序錯誤（Timing Errors）： 這種錯誤可能由於計算過程中的計時不準確，導致門操作和量子位元的不正確同步。

測量錯誤（Measurement Errors）： 在量子計算結束時，測量操作可能受到干擾，導致計算結果的錯誤。

這是目前IBM Heron r1 processer的error rate map

A screenshot of a computer

Description automatically generated

From: https://quantum.ibm.com/services/resources?tab=systems&system=ibm\_torino

CZgate是這個quantum processer主要使用的two-qubit gate，錯誤率的中位數是0.004101，由於量子錯誤不僅僅只有bit-flip error，還有phase error，這0.004101的錯誤率在經過幾十次以上的運算後就會大到無法得到正確答案。

所以量子糾錯演算法對於量子電腦的成功運行是非常重要的。量子糾錯算法不僅要考慮糾錯之後的正確率，糾錯所用到的two-qubit gate還要符合量子電腦qubit網格的排列方式。

量子糾錯算法的設計不僅僅於程式方面，一個大型的量子電腦系統在開發時也要考慮如何優化量子電路布局使量子糾錯算法更容易實作。

附圖Google Sycamore processor

A colorful cross pattern

Description automatically generated with medium confidence

Form:https://blog.research.google/2019/10/quantum-supremacy-using-programmable.html

### 第二章 糾錯演算法的實作與分析

### three-bit repetition code

### 將一個待糾錯的qubit運算重複三次，理應獲得三個一模一樣的結果，然而運算過程可能會出錯，導致三個qubit不全相同。three-bit repetition code即為將這三個qubit統一更正為佔多數的答案，意即將此三qubit放入全加器並取carry位。

### three-bit repetition code運作如下範例：

### 1 -> 111

### 0 -> 000

### 000,001,010,100 -> 0

### 111,110,101,011 -> 1

### 下圖為傳統電路的全加器與真值表

### A diagram of a machine Description automatically generated with medium confidence

### A diagram of a network Description automatically generated with medium confidence量子電路實作全加器

### input：q0,q1,q2

### output：q3(carry)

### 再將結果寫回q0：

### A diagram of a diagram Description automatically generated

### 環狀分組糾錯

### 將一個待糾正的qubit運算進行六次，並運用六個結果循環分組進行糾錯。依序將012,123,234,345,450,501分為一組形成three-bit repetition code並執行糾錯。

### 6-qubit環狀糾錯成果

### A graph with green and red lines Description automatically generated

### 成效不彰，有時正確率甚至不如糾錯前，如上圖。

### 錯誤情形

### 000101 -> 000000

### 000110 -> 000110 (Error)

### 假設目的是將所有qubit糾正為0，則當遇到連續兩個1時會發生死循環，無論糾錯幾次都無法修正為0。意即，只要有連續的錯誤就是就會沒辦法修正，連續兩個qubit同時壞掉是很容易發生的，尤其cx-gate(或其他雙qubit gate)。為解開死循環，須將取樣順序打亂。

### 如何取樣

### 經實驗發現，依照Index：[-2,0,1]取樣的糾錯效果最佳。假設欲修正index為4的qubit，則取index為2(4-2)、4(4+0)與5(4+1)的qubit組成three-bit repetition code，獲得糾錯結果後寫回index為4的qubit。

### A black background with white text and numbers Description automatically generated

### A graph of a number of different colored lines Description automatically generatedA graph of a number of people Description automatically generated with medium confidence

6-qubit環狀糾錯

10-qubit[-2,0,1]

10-qubit[-2,0,1]

未糾錯

6-qubit環狀糾錯

10-qubit[-2,0,1]

20-qubit[-2,0,1]

未糾錯

### 量子電腦的錯誤率很關鍵

### 以下兩圖分別展示當量子電腦本身的錯誤率為0.002以及0.007時的糾錯成果，可見當量子電腦本身的錯誤率為0.007時效果遠不如錯誤率為0.002的量子電腦。由此可知，量子電腦本身的錯誤率很大程度的影響糾錯演算法成功率。

### 新的死循環

### 假設欲將6-qubit以index=[-2,0,1]取樣糾錯以修正為全0，而當錯誤情形如下時，越是糾錯會越錯越多。

### 100001

### ->100001

### ->110001

### ->111001 …..

### 原因是若第一組three bit repetition code糾錯出來的結果仍是錯的，寫回第0-qubit後，會在被第二組three bit repetition code取樣進行糾錯。這顯然將導致一步錯，步步錯的結果。

### 解決方法

### 更好地辦法應是，將每一組three bit repetition code糾錯出來的結果先暫存至另一組6-qubit相對應index的qubit中，取樣下一組three bit repetition code時取的都是原始未糾錯過的，以避免每次糾錯互相影響。待所有糾錯都完成，再將結果寫回原本的q0。

### date qubit: 100001

### support qubit:000000 calculate->

### date qubit: 100001

### support qubit:000001 copy(write back)

### date qubit: 000001

### support qubit:000001

### 以下兩圖分別為錯率0.002與0.007的糾錯結果：

### A graph of different colored lines Description automatically generated

### A graph of different colored lines Description automatically generated

### 可以發現錯誤率稍微變大，糾錯率就大幅降低。

### 隨機打亂

### 同樣錯誤率為0.002與0.007的糾錯結果，增加了另一種作法的觀察——模擬qubits隨機重新排列(灰色)。

### A graph of different colored lines Description automatically generatedA graph of a graph Description automatically generated with medium confidence

### 雖然現實中不存在這種隨機打亂的方法，但可以觀察「隨機打亂」是否有助於大幅提升糾錯能力，結果發現確實是有助於的。

### ->前面說的兩個問題()都很重要，且我們需要一個(接近)隨機打亂的方法，在每次糾錯前執行。

### 理想中的隨機打亂

### 下圖以IBM brisbane的量子電腦的qubit排列方式為例，示範隨機打亂的步驟：

### A diagram of a data system Description automatically generated with medium confidence←圖一

### 圖一：橘色框內的qubit為待糾錯的資料，紅色框起的qubit為欲儲存糾錯結果的額外qubit空間。圖中橘框內編號40、41、42為相鄰qubit，首先取這三個qubit進行糾錯(放進加法器取carry位)，將結果存放到編號60的qubit。同理，取編號44、45、46進行糾錯並將結果存放到編號64的qubit。

### A diagram of numbers and a number Description automatically generated with medium confidence A diagram of numbers and a number Description automatically generated with medium confidence

### 圖二↑ 圖三↑

### 圖二、圖三：編號53、54已存放糾錯後的結果後，將待糾錯資料的qubit兩兩位置交換，交換方式如圖二的藍色框線所示，完成以後再如圖三的橘線所示兩兩互換位置，即完成一次打亂。打亂後得到新的編號40、41、42的qubit，取此三個qubit進行糾錯得結果存入編號60的qubit。以次類推，每次打亂後取編號40、41、42的qubit糾錯，將結果依序填入下方儲存空間，直至空間填滿，即獲得所有糾錯結果。

### 每次打亂後取的qubit情形舉例：

### [(1, 3, 25), (1, 5, 7), (1, 23, 27), (2, 4, 29), (3, 5, 27), (3, 7, 29), (4, 6, 7), (5, 6, 29), (8, 9, 13), (8, 10, 11), (8, 12, 14), (8, 20, 28), (9, 10, 12), (9, 18, 20), (10, 12, 17), (10, 14, 16), (10, 21, 28), (11, 16, 18), (12, 14, 15), (12, 16, 18), (12, 19, 21), (13, 14, 16), (14, 17, 19), (14, 18, 20), (15, 16, 17), (16, 20, 28), (18, 21, 28), (19, 20, 21), (22, 25, 27), (23, 24, 25)]

### A graph of a graph Description automatically generated with medium confidence不同錯誤率下的糾錯率

### 註解：在A graph of a line Description automatically generated with medium confidence500個cnot gate時，我們人工加入了一些很大的錯誤率，用來測試這些糾錯方法是否能將正確率拉回正常水平。

### 代號解釋：

### 6 bit：最基礎的6bit環狀糾錯

### 20bit\_rand：取樣index為[-1,0,2]的方法

### 20bit\_r\_d：取樣index為[-1,0,2]，且將結果複製出來再寫回，而不是使用糾錯結果對後續qubit糾錯

### 40bit\_d\_p：算完隨機打亂再寫回

### 21bit\_R：使用我們提出的方法在IBM量子電路上模擬

### 30bit\_R：使用我們提出的方法在IBM量子電路上模擬

### 如何處理相位的問題:

### 可以將要修正的相位先旋轉映射到x軸上，糾錯算法完成後再返向旋轉回去，以+H、-H為例，0、1和+H、-H互為經過Hadamard gate的結果，若要做+H、-H的修正可以先用Hadamard gate轉變為0、1的狀態，修正完成後再利用Hadamard gate轉回為+H、-H。

### 取錯誤率的理由:

### 我們使用三個錯誤率做實驗，分別為0.007395、0.002795、0.001433，分別為ibm\_ Eagle r3的Median ECR error、ibm\_Egret的Median ECR error和ibm\_Egret的Minima ECR error，第一個是我們用來參考現在(2024/2)能用到的量子電腦平均錯誤率，第二個跟第三個是IBM開發中的量子電腦錯誤率，我們想要用此推估這個糾錯演算法在未來的淺力。

### 模擬器CODE

### 

### 參考文獻

### <https://zh.wikipedia.org/zh-tw/%E9%87%8F%E5%AD%90%E9%96%98>

### <https://www.researchgate.net/figure/a-Toffoli-gate-b-Swap-gate-c-Quantum-gate-d-Quantum-gate-stored-in-a-superposition_fig13_257641933>

**圖目錄**

**表目錄**