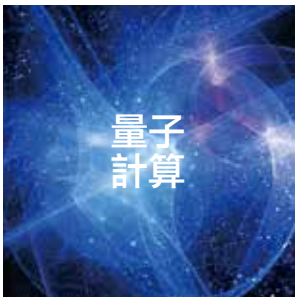

量子演算法

作者：張慶瑞

作者簡介

張慶瑞是臺灣大學物理學系特聘教授。

割之彌細，所失彌少，割之又割，以至於不可割，則與圓周合體而無所失矣。—— 三國魏國，劉徽（225 年～ 295 年）

古典計算就像獨奏的聲音，只是美妙音符的單純串接。量子計算就像一首交響樂，眾多天使樂章相互和鳴。—— 美國，賽斯・洛依德（Seth Lloyd，1960 ～）

綱要：

第一節：甚麼是演算法

大家常用的 YouTube, Instagram, Facebook 等，或者是手機各種 App 軟體，不同業者都有自己獨門的演算法與特殊功能。使用者常會覺得某些軟體好用或是特別友善，主要的差別就是其中演算法好壞。但演算法到底是甚麼？演算法是解決定特定問題的有效且按部就班的實施步驟，小學時常用的輾轉相除法就是一種標準演算法，而電腦則是可以執行任何演算法的通用性機器。英文名稱「Algorithm」演算法出自「Algoritmi」，是 9 世紀偉大波斯數學家花拉子米（Al-Khwarizmi）的拉丁譯名。「花拉子米法」利用「移項」與「消去」的制式步驟解出一元二次方程式，這些序列步驟在現代稱爲「程式」或是「演算法」。演算法在中國古代則稱爲「術」，最早出現在《周髀算經》、《九章算術》。特別是《九章算術》，給出四則運算、最大公約數、最小公倍數、開平方根、開立方根，線性方程組求解的演算法，而三國時代的劉徽割圓術給出計算圓周率到任意精確度的疊代程序的求圓周率的演算法。但是只有演算法沒有實施演算法的良好工具是無法完成眞正工作，所以劉徽當時利用他的極限演算法只能先分割圓爲 192 邊形，並估計出圓周率 π 爲 $157/50 = 3.14$ ，再計算出正 3072 邊形的面積，求得圓周率 $\pi = 3927/1250 = 3.1416$ ，稱爲徽率。劉徽的九章算術注文明白寫著：「割之彌細，所失彌少；割之又割，以至於不可割，則與圓周合體而無所失矣。」劉徽知道極限的概念，也

作者感謝卓建宏整理部分資料與校稿，卓妙如繪製書中圖畫。

知道以正多邊形繼續分割圓可以更準確逼近 π 值，但是他沒有適當工具可以眞正發揮割圓術到極限，電腦的出現讓各種演算法得到眞正舞台得以發揮。

勒芙蕾絲（Ada Lovelace Byron）是眞正認識電腦潛能的第一人，主張電腦不只可以像手指或是算盤計數，也能解複雜的運算，她也是最早的程式設計師，曾經發表解伯努利微分方程的演算法。演算法是電腦處理資訊的重要步驟，用來執行一個指定的任務，可以簡單到計算薪水，列印各種報表，也可以複雜到解積分方程甚至協助做出決策。當演算法處理資訊時，會先從輸入裝置讀取資料，利用演算法的指令序列處理資料，然後將結果寫入輸出裝置。電腦沒有演算法，不管何種電腦都無法處理任何事情。簡單說，演算法是由有限序列指令所構成，用來解決特定的問題。如果今天有個問題是煮冷凍水餃，演算法就是如何煮冷凍水餃的食譜程序，需要經過如下步驟：

A 準備輸入資訊

步驟 1. 由冰箱拿出冷凍水餃

B 準備適當電腦的執行工具

步驟 2. 準備煮鍋

C 演算法

步驟 3. 放入足夠水，打開爐子大火加熱至水滾沸後

步驟 4. 將冷凍水餃放入鍋中

步驟 5. 待水再度滾沸後加入冷水少許，此步驟重複三次

D 結果輸出

步驟 6. 將煮熟水餃放入餐盤中

步驟 1 就像是電腦的輸入裝置，**步驟 6** 就是輸出裝置，而**步驟 2** 就是選擇執行演算法指令的硬體系統，由於加熱效果與容量不同，選取平鍋或是圓鍋也會導致水餃的口味不同，這也是工欲善其事，必先利其器的道理。**步驟 3** 至**步驟 5** 就是煮水餃的演算法，其中**步驟 5** 更是煮出好吃水餃的重要遞迴步驟，重複點水三次的目的是透過冷水讓水餃的溫度下降。降溫有兩個目的，一個是希望均勻加熱，避免外部餃皮過熟而內部餃餡仍然未熟。另一個原因則是避免餃皮因高溫糊化，溫度略爲下降可使餃皮更有彈性而增加口感。只要演算法足夠清楚與有效，任何人都可以依據演算法做出好吃的水餃。很多人會煮水餃也知道要點水三次，但卻不理解爲何要做此重複步驟，這也是演算法的精髓所在，讓大眾在「知其然不知其所然」狀況下都可做出完美的結果。

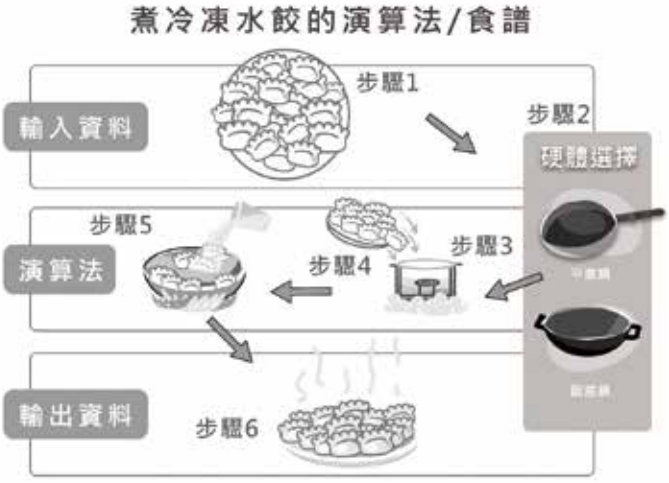

圖 1：煮冷凍水餃的演算法與執行演算法的鍋具選擇。

第二節：量子演算法

演算法是解決問題的邏輯思維與步驟，但是也會依據工具與環境的不同而適當調整步驟。在有小圓格的煎鍋上很容易煎出圓形荷包蛋，但是在大平鍋上的荷包蛋形狀就可能常是多邊形。古典演算法是在古典電腦上執行，利用有限的指令序列與步驟用來解決問題，同樣的，量子演算法也需要在量子電腦上執行。量子電腦和古典電腦最主要的差異在**古典位元與量子位元**，例如量子疊加，量子糾纏甚至量子量測這些特性在古典電腦中並不存在，因此量子演算法與古典演算法之邏輯思維完全不一樣。量子演算法中除了有量子電路模型，還有像是量子絕熱等的模型（Quantum adiabatic computation），都可以實現比古典演算法快速的量子演算法。爲了能充分利用這些特殊量子性質，如何設計出實用且有效率的量子演算法就成了目前相當重要的課題。目前已經知道，在某些問題上量子電腦相較於古典電腦具有相當大的優勢，例如質因數分解，就可以利用量子演算法將計算時間大幅減少，把需要數百萬年才能計算的問題轉變成有限時間內就能得解決的問題。量子電腦只是工具，量子演算法就是在適當工具的操作步驟，再好的機器也必須要有完美的演算法才能眞正發揮功能。量子演算法與量子電腦就像形與神，缺一不可，只有量子電腦沒有量子演算法，就像〈愛拚才會贏〉的歌詞，「無魂有體親像稻草人」，不會發揮太多作用。量子計算科學家們的首要任務之一，就是要針對個別問題找出量子電腦的最佳的運行步驟，藉此解決目前古典電腦的困難問題。下面會分成三部分來介紹量子演算法，包括多伊奇／喬薩演算法 ❶，量子－古典混合式演算法以及可執行於具有容錯功能 ❷ 的量子電腦上的演算法。

一、多伊奇 / 喬薩演算法

多伊奇（David Deutsch）1985 年處理的一個位元（即 0 或 1）輸入黑盒子機器（數學上就是 f 函數）後的輸出問題（即 $f(0)$ 或是 $f(1)$），後來在 1992 年由多伊奇與喬薩（Richard Josza）一起推廣到多個位元在黑盒子的輸出問題，展示了量子計算能力的強大加速能力。儘管到目前爲止這個演算法幾乎沒有任何實際用途，但是比起任何古典算法，多伊奇／喬薩演算法是第一個展示有指數級加速作用的量子演算法，也開啟了其他量子演算法的應用研究方向。這裡僅用非數學的直覺說法來闡釋伊奇／喬薩演算法。

上面多伊奇的數學問題可以用下圖的古典色球售貨機（即 f 函數）來理解，如下圖將一枚硬幣投入售貨機中就會滾出一顆球，如果手中有一個銀幣（即 0）及一個金幣（即 1），而售貨機中滾出球的顏色可能是紅色（即 0）或藍色（即 1），金幣與銀幣分別投入售貨機後，滾出來紅球或藍球的機會相同，而且一枚硬幣只能購買一顆色球。則售貨機在金幣與銀幣可能會（甲）都會購得相同顏色的球，或是（乙）分別購得不同顏色的球？對於古典

❶ **註**：Abbott, A. A.,"The Deutsch-Jozsa problem: De-quantisation and entanglement", *Natural Computing*, 11 (2012), 3~11。

❷ **註**：可容錯（fault-tolerance）：指的是當電腦的位元發生錯誤時，電腦本身有自我修改錯誤的功能。

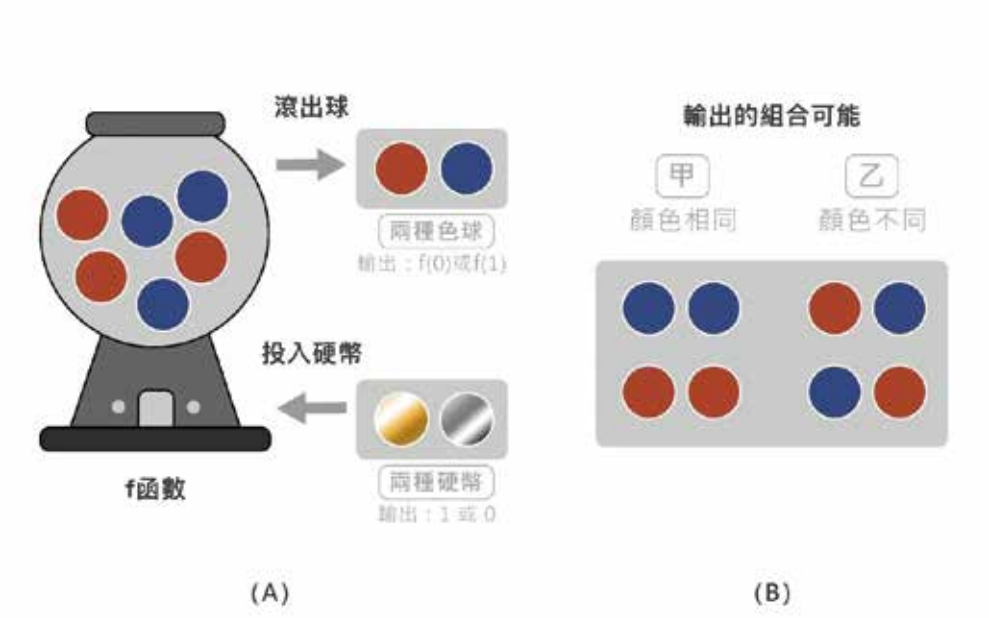

圖 2：（A）兩枚不同硬幣分別投入古典售貨機，$f(x)$，一枚硬幣投入後可滾出一枚色球，售貨機內裝有紅與藍兩種色球，這種就是多伊奇的一個位元輸入黑盒子機器後的輸出的映射問題。（B）兩枚不同硬幣投入後，滾出來的四種色球組合狀態，可以分為兩類：同樣顏色色球的常數態，或是不同顏色色球的平衡態。古典電腦計算時，必須投入兩次硬幣後才能知道售貨機滾出色球組合結果，但多伊奇／喬薩量子演算法可以只投入一枚硬幣就知道組合結果。

電腦，只有將兩個硬幣都投入售貨機後才可能知道最後滾出的球是下圖中的（甲）還是（乙）的狀態，數學上稱（甲）狀態爲常數（constant，多對 1 映射），而（乙）狀態爲平衡（balance，1 對 1 映射）。

多伊奇演算法巧妙的利用量子力學的疊加與糾纏特性，只投一枚量子硬幣就可以解決這個問題，下面是多伊奇演算法的步驟，這裡我們不嘗試證明數學，而是只單純敘述演算法的邏輯。多伊奇首先在投入售貨機前，先在手上的金幣與銀幣上操作第五章中所提及的阿達瑪閘（H 閘），以 H 閘將金幣與銀幣分別轉換成有疊加態的「量子硬幣」。量子電路中的 H 閘的功能可以想像成一種古典硬幣的修改工具。當在古典硬幣上使用這個工具時，古典的金幣與銀幣就會成爲了量子硬幣，是銀幣和金幣的組合。量子硬幣有兩個特徵是：（1）一旦量測量子硬幣時，量子硬幣各有 50% 的機會被發現是古典金幣或古典銀幣，（2）在量子硬幣上，只要再使用一次 H 閘時，量子硬幣就會還原成古典金幣或銀幣狀態。

多伊奇也把古典售貨機（f ）也轉化成一種特殊的量子售貨機（ U_f ），量子售貨機裏有內建一個 CNOT 閘，這 CNOT 閘的功能很特殊，會創造糾纏性，使得量子金幣的量測結果會影響古典銀幣的色球輸出。當量子金幣與量子銀幣分別投入量子售貨機時， U_f 機器出口分別滾出一個量子金幣，及另一個量子金幣經過量子售貨機後的輸出色球組合

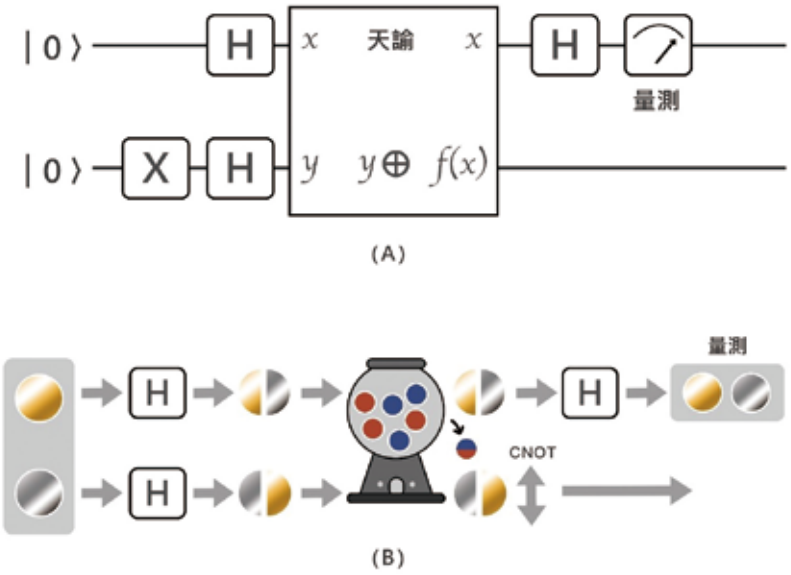

圖 3：多伊奇的 U_f 量子售貨機的特殊功能在輸出右下方的 $y \oplus f(x)$。（A）多伊奇量子演算法的量子電路圖，可以一石二鳥，用量測一個量子硬幣決定兩個古典硬幣的輸出組合。（B）金幣與銀幣的量子電路中轉變過程示意圖，CNOT 閘在量子售貨機中的功能類似古典邏輯電路中互斥或閘的功能 3 。

與投入量子銀幣的糾纏結果。如圖 3，把有疊加態的量子金幣與量子銀幣都投入量子售貨機後，直覺的知道，由於量子金幣具有古典金幣與古典銀幣的疊加態，滾出的色球組合也應是紅球與藍球的疊加態。當最後對投入的量子金幣的再做一次 H 閘後，因爲 H 閘特殊性質，量子硬幣會還原回到古典金幣或銀幣狀態。這時如果量測到結果的是古典金幣，則分別投入古典金幣與古典銀幣到如圖 2 的古典售貨機內，都會滾出紅球或藍球，也就是（甲）的常數狀態；反之如果量測量子金幣中得到的是古典銀幣，這時分別投入古典售貨機的古典金幣與古典銀幣則會得到不同顏色的球，即爲（乙）的平衡狀態。多伊奇演算法在只量測一個量子硬幣的狀況下就確定了古典售貨機滾出兩個色球的組合狀態。

多伊奇演算法等價於利用量子的疊加與糾纏特性創造出一個希爾伯特空間中嶄新的量子售貨機，只要量測一枚投入的量子硬幣，古典機器中滾出色球的組合結果也立刻便可確定。這種硬幣投入售貨機後滾出色球的現象，在數學上就稱爲映射，而明顯的，在架構出恰當的量子售貨機（ U_f ）的形式後，多伊奇演算法確實比古典演算法可以使用更少計算過程就可得出映射後的結果。多伊奇演算法跟古典演算法的一個硬幣滾出一個色球的做法相比之下，就是利用量子特性開發出了有疊加態的量子硬幣，與內設糾纏特性的量子售貨機（ U_f ），進而達成一石兩鳥的聰明作法。多伊奇／喬薩演算法就是將此方法擴展到 N 個硬幣的狀況，也就變成更厲害的一石 N 鳥的指數加速打獵法。

二、混合式量子演算法

1930 ～ 1940 年代是計算機科技萌芽發展的時期，當時的古典電腦體積龐大且計算效能不高，目前的量子電腦也有類似的缺點。量子電腦現在只有 127 多個物理量子位元（physical qubit），能處理的問題規模不大，又不能自我糾錯，距離容錯通用

3 註：受控反閘，參見 https://zh.wikipedia.org/wiki/受控反閘。

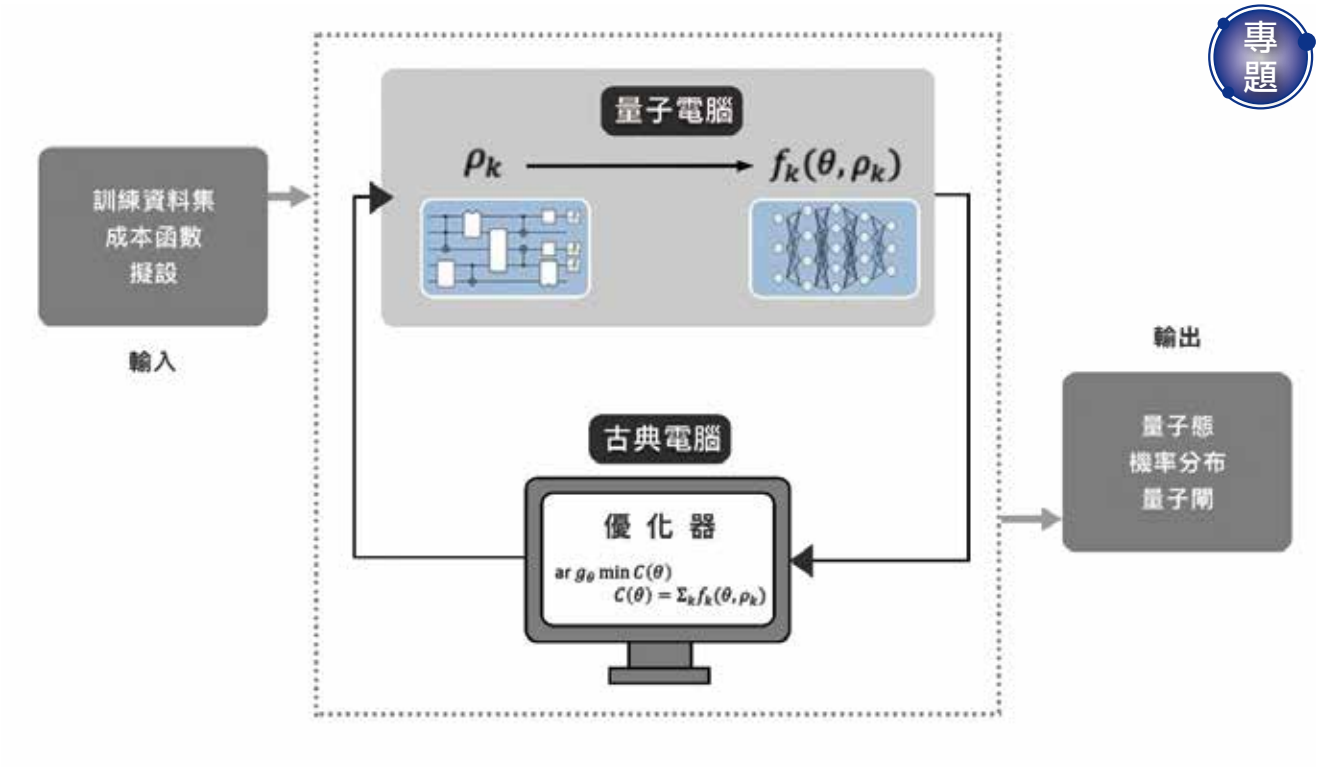

圖 4：古典電腦與量子電腦整合結構示意圖。

型量子電腦仍有相當大差距。雖然目前量子電腦有許多缺陷，但仍可利用小規模量子電腦與古典電腦互相搭配以進行計算。利用分工的概念，讓量子電腦和古典電腦彼此截長補短。目前量子電腦中的量子資訊會隨時間增加而自動消散，並使得錯誤率逐漸增加，因此無法進行大型而長久的量子運算。另一方面，古典電腦雖然錯誤率很低又可自我糾錯，然而由於古典位元本身限制，無法有效率的計算龐大變數且大型的問題。因此結合古典與量子電腦，就是近期 NISQ 的主流方法。出名的變分量子演算法（variational quantum eigensolver，VQE）正是屬於這種混合式演算法的架構。

你可曾有想過吃進去的食物究竟是如何消化成身體可以吸收的形式？又或者是保健食品的酵素究竟是如何促進體內機能運作？身體中各式各樣的化學反應牽涉到不同分子之間的電子轉移、化學鍵的生成與破壞，所有生化過程都是複雜的量子作用，也只有量子力學才能描述生化的動態過程。但身體中所有反應至少牽涉上百甚至上千的原子與分子參與，當反應規模過大時，古典電腦是無法正確模擬處理，因此使用量子電腦進行相關量子物理化學模擬是不二選擇。描述一個物理系統需要哈密頓量（Hamiltonian）[4]，哈密頓量裏會包含整體系統的所有資訊，動能、位能以及不同粒子之間的相互作用。當所有電子的狀態處於最低能量態時，此時分子會處於最穩定的狀態，但若分子吸收外加能量（例如：電磁波），分子裏的電子會躍遷到激發態，形成較不穩定的狀態。計算這些原子與分子資訊可以幫助科學家掌握化學分子的特性及相關化學反應的路徑，大量縮短藥物開發及製藥等的時程，而 VQE 正是處理這方面問題的特殊演算法。

在變分量子演算法的架構中需要量子電腦和古典電腦的相互配合，量子電腦負責量子態的製備與演算，而古典電腦則是負責參數的優化與量子態輸入與輸出準備。變分演算法的工作流程分成量子電腦

[4] 註：哈密頓量是一個物理辭彙，也是一個描述系統總能量的算符，以 H 表示。

與古典電腦兩個部分。將系統的哈密頓量以二次量子化（second quantization）[5]的形式寫下包含各個電子與原子核之間的交互作用，但在實際情況中，分子中不同的電子與原子核彼此會互相作用與影響，幾乎不可能解出精確的原子核與電子的波函數形式，因此需要使用一些近似來簡化系統的哈密頓量[6]。使用量子閘先建構出分子系統的擬設態（ansatz），擬設態內含有一些可供古典電腦最佳化的調整參數，接著就由量子電腦與古典電腦的相互配合來計算出分子的基態能量。變分量子演算法其實就是用一個古典優化器來訓練一個量子線路機器的過程，而如此透過古典優化器來得出模型參數的過程和機器學習的過程頗爲相似。我們先建造出擬設態，在計算過程中，我們利用古典電腦對量子電腦進行優化的訓練，一直調整到最佳的結果產生出來。

三、可容錯量子電腦上的演算法

(一) 秀爾演算法

隨著通訊科技與網際網路的興起，資訊的分享無時無刻都在全球各的間發生，大大的促進人類生活的便利性。疫情時期，由於封城，員工多數居家隔離工作，只需透過加密軟體輔助，我們可以從遠端進入公司的電腦繼續工作。在家想購買網路各種食物及用品，只要透過網際網路的下單及線上認證與刷卡的方式即可輕鬆在家等貨物上門。但這些都需要通訊安全的措施以確保個人資訊不被竊取，因此各種加密技術也應運而生，目前公認最難破解的加密技術之一當屬 RSA[7]非對稱式加密演算法。

小美要傳送機密文件給在美國的小明，如何確保文件不被他人竊取呢？現行加密系統分成兩大類：對稱式加密和非對稱式加密。對稱式加密是小明和小美共同協定一把密鑰。在小美發送文件之前，先使用密鑰將機密文件轉譯成密文，以確保傳遞過程的安全性。當這封加密文件傳遞至小明時，小明使用相同密鑰對密文進行解譯即可。小美和小明也可以使用非對稱式加密，就是加密和解密的過程分別使用不一樣的鑰匙。公鑰用於加密而私鑰用於解密，因而稱其爲「非對稱式」。小明先製備公鑰與私鑰各一把，公鑰可以公開，而私鑰必須由個人妥善保管。當小美用公鑰將文件轉譯成密文後，小明只需要藉由私鑰將所收到的文件轉譯回來即可。對稱式加密和非對稱式加密各有優缺點，對稱式加密中雙方密鑰的保管及傳遞是重要的，傳遞密鑰的通道不夠安全則可能被外人竊取。但儘管非對稱式加密沒有密鑰傳遞安全性的問題，但因爲通常加解密過程中會牽涉到更複雜的數學計算，因此效率不如對稱式加密快速，在實際情況中，兩者常互相搭配使用。

RSA 加密是非對稱式加密，其可靠性是因爲極大整數的質因式分解非常困難，但秀爾演算法[8]的出現卻使得非對稱加密的通訊隱私變得不安全。1994 年秀爾提出計算整數質因數分解的量子演算法，相較於古典演算法而言確實具有指數加速性。秀爾演算法證明量子電腦的確能有效加速因式分

[5] 註：表示成這樣的形式是較能體現出多電子體系的波函數行爲。
[6] 註：考慮玻恩／歐本海默近似法（Born-Oppenheimer approximation）。
[7] 註：RSA 非對稱式加密演算法是由三位科學家姓名所組成的：李維斯特（Ron Rivest），沙米爾（Adi Shamir），艾得曼（Leonard Adleman）。
[8] 註：秀爾演算法，參見 https://zh.wikipedia.org/wiki/秀爾演算法。

圖 5：對稱式與非對稱式加密系統示意圖。

解，讓原本在傳統電腦要耗費上百年的運算時間的問題，如今量子電腦有可能在幾秒內計算完成，也因此開啟量子計算的應用大門。秀爾演算法主要分成兩個部分：第一部分需先使用古典電腦將質因數分解問題轉化成尋找週期的問題（period-finding subroutine），接著第二部分再使用量子電腦進行尋找週期的計算，其中尋找週期演算法的核心就是量子傅立葉變換（quantum Fourier Transformation，QFT）。量子傅立葉變換 [9] 本質上就是一般數學中的離散傅立葉變換 [10]，可以透過下圖來直觀的理解傅立葉變換的概念，空間中的波形可以將看成不同比例的正弦波相加，也可以用頻率空間來描述這個波形。如下圖所示，一個紅色的複雜波可以被分解成不同比例的黑色正弦波與紫色餘弦波的組合，這就是傅立葉變換。由於量子態中有波動特性，藉由波動中建設性及破壞性干涉，可以快速找出最佳結果，相較於古典電腦計算有指數型的加速。

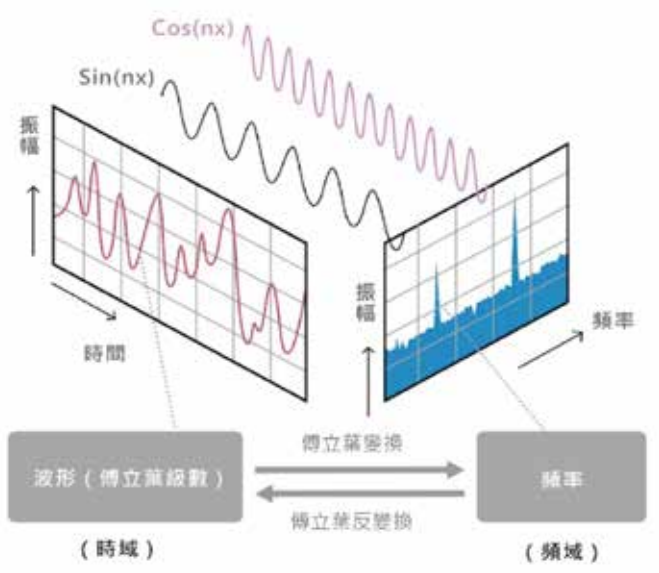

圖 6：傅立葉變換的分解示意圖。紅色的波可以分解成不同比例的黑色正弦波與紫色的正弦波的組合，這就是傅立葉變換。由時域向頻域的轉換稱為傅立葉變換，反過來由頻域向時域的轉換則稱為傅立葉反變換。

若我們想要使用秀爾演算法找出一個整數 $N = pq$ 的質因數分解，其演算法步驟可以概要的拆成幾個步驟：

步驟 1. 先挑選一個小於 N 的正整數 a ，先驗證 a 是不是 N 的因數：若為 N 的因數，則停止整個演算法步驟，代表我們恰好找出 N 的其中一個因數，而另外一個因數也能隨之得出。但一般運氣不

[9] 註：量子傅立葉變換，參見 https://zh.wikipedia.org/wiki/傅立葉轉換。
[10] 註：離散傅立葉變換，參見 https://zh.wikipedia.org/wiki/離散傅立葉轉換。

可能如此好，a 若不是 N 的因數，則必須開始尋找 a^r 除以 N 餘 1 的週期 r 。數學符號習慣寫成如下， $\mathrm{mod}N$ 就是 a^r 除以 N 的餘數，所以我們要找 $a^r \equiv 1(\mathrm{mod}N)$ 。

步驟 2. 這個尋找週期 r 的步驟必須使用量子傅立葉變換，也是秀爾演算法裏眞正量子加速的計算過程。量子傅立葉變換能夠較快找到讓上式成立的最小 r 值。

步驟 3. 當找到 r 後，就代表 a^r-1 是 N 的整數倍，也就是

$$\begin{aligned}(a^r-1) &= \left(a^{\frac{r}{2}}-1\right)\left(a^{\frac{r}{2}}+1\right) \\ &= Nd = (pq)d \ ,\end{aligned}$$

其中 $N=pq$ ，而 p 和 q 是 N 的質因數，而 d 是正整數。此時會有以下幾種可能：

甲

$$\left(a^{\frac{r}{2}}-1\right)=N,\ \left(a^{\frac{r}{2}}+1\right)=d,$$

或是

$$\left(a^{\frac{r}{2}}-1\right)=d,\ \left(a^{\frac{r}{2}}+1\right)=N;$$

乙

$$\left(a^{\frac{r}{2}}-1\right)=p,\ \left(a^{\frac{r}{2}}+1\right)=qd,$$

或是

$$\left(a^{\frac{r}{2}}-1\right)=q,\ \left(a^{\frac{r}{2}}+1\right)=pd;$$

丙

$$\left(a^{\frac{r}{2}}-1\right)=pd,\ \left(a^{\frac{r}{2}}+1\right)=q,$$

或是

$$\left(a^{\frac{r}{2}}-1\right)=qd,\ \left(a^{\frac{r}{2}}+1\right)=p;$$

若 $\left(a^{\frac{r}{2}}-1\right)$ 及 $\left(a^{\frac{r}{2}}+1\right)$ 都無法被 N 整除，也就是乙 與 丙 的狀況時，則代表 $\left(a^{\frac{r}{2}}-1\right)$ 及 $\left(a^{\frac{r}{2}}+1\right)$ 其中必定各含有一個 N 的因數，這時再使用輾轉相除法可求出 N 的質因數分解的 p 和 q 。

若上述步驟無法成功的找到 N 的質因數分解，代表所選取之 a 是不對的，則重新回去**步驟 1** 挑選其他的正整數 a，並且重複 **1, 2, 3** 的步驟。

秀爾演算法是標準的古典與量子演算結合的混合式演算法，只有**步驟 2** 用到量子傅立葉變換來加速尋找週期的過程。秀爾演算法眞正計算加速的步驟就是利用量子傅立葉變換來快速尋找未知週期 r ，在古典演算法中因爲沒有量子位元的相位可以操作，因此找尋未知週期是相對緩慢的。下面用一般人可理解方式來解釋神奇的 QFT 找尋週期的邏輯[11]。

地球的日夜週期是 24 小時，但每個人的生理時間則因人而異，如果有一個人在暗無天日的地穴中生活，他每天生理時鐘是 23.5 小時，也就是每天都非常規律的以 23.5 小時的作息周而復始。地穴的牆上掛了幾個不同週期的鐘，每天早上起床時會看到許多不同週期時鐘在牆上，但不同的時鐘反而讓他完全困惑，也無法知道那個時鐘的時間才能正確描述他的生活週期。他思考如何快速找出正確週期的時鐘？想了許久，終於想出一個聰明方法，在每個時鐘下貼上一張便條紙，並在便條紙正中央插上一根大頭針，每天起床後就將時鐘下的便條紙上的大頭針沿著時針方向移動一公分。幾天之後，只有與生活作息一樣的時鐘下的便條紙上的大頭針會沿同一方向運動而很快離開便條紙區域，其他週期不相同的時鐘下的便條紙上的大頭針只能在紙上從事隨機運動而無法離開便條紙。

[11] **註**：參見 https://www.scottaaronson.com/blog/?p=208。

上面的找尋正確時鐘的過程基本上就是量子傅立葉變換。更準確說，QFT 就是一種線性變換，將一個複數向量映射到另一個複數向量。輸入向量就類似於每早起床看到的時針指向，而輸出向量則記錄了大頭針在便條紙上位置的軌跡向量。所以不同的時鐘下的便條紙的軌跡與時鐘的時針指向之間是一個線性變換，這個線性變換把一序列的指針指向資訊映射成指針的週期。另一種理解方式是相位干涉，這也是量子力學與古典機率論間最大不同之處，雖然機率值總是非負值，但量子力學中的機率振幅可以是正的、負的，甚至是複數。因爲如此，與特定答案的不同的振幅便會出現「破壞性干涉」並相互抵消，這也是與生理週期不同的時鐘下的便條紙上的大頭針無法離開便條紙的原因。「破壞性干涉」正是秀爾演算法中發生的事，重點在於，與眞正週期不同的其他週期影響，都會因爲破壞性干涉而抵消。只有與眞正的週期相符，因爲每次指向同一方向才會出現建設性干涉，所以在最後進行測量時，會很容易找到眞實週期。

(二) 格羅佛演算法

最後要介紹另一個著名的量子搜尋演算法——格羅佛演算法，這個演算法是對無序的資料進行有效率的搜尋。如果桌面上擺了 n 本書，而你要的那本書就在其中，但每次只能翻開一本書來確認，那需要翻幾次才能找出這本書呢？使用一般古典電腦演算法最多會需要翻 n 次，平均需要翻大約 $n/2$ 次才能找出所要的書，也就是 $O(n)$ 的複雜度才能找到所要的結果。格羅佛演算法透過量子演算的加速後，僅需要 $O(\sqrt{n})$ 的複雜度就可以找出所要的書。

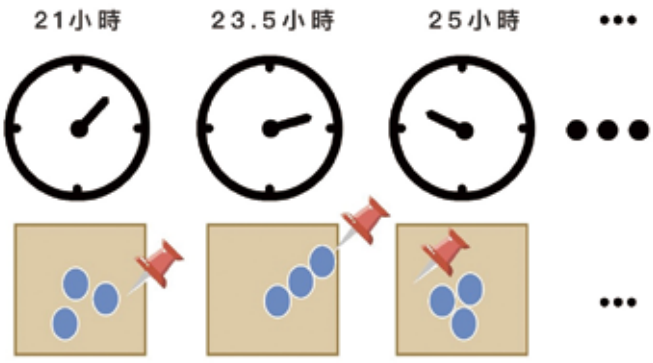

圖 7：地穴中牆上有許多時鐘，只有一個鐘與生活作息的週期相同，如何快速找到正確週期的時鐘？每天早上起來，只要把便條紙上的大頭針沿著時間方向移動一公分，很快就可以知道 23.5 小時是正確描述生活週期的時鐘，因為只有相同週期的大頭針產生的路徑才會出現建設性干涉。

除了書本搜尋問題外，生活中還有許多問題都可以使用量子搜尋演算法來加速，如著色問題、商店最佳場地選址等問題皆可以使用格羅佛量子演算法幫助我們找出答案。搜尋的複雜度從 $n/2$ 次到 $\sqrt{n}$ 次，在 $n = 100$ 萬的系統，古典搜尋需要平均 50 萬次試錯，但用量子演算法只需要 1000 次，這種差別在 n 越大的系統就愈省時間。在某個大都市裡，得益於觀光與經濟的快速發展，使得都市因而快速的繁榮起來，市長希望能多招募一些便利商店進駐來活化都市生活機能。由於都市裡能開店的地點有限，選址的好壞將影響到未來的營運，對於店家數目非常多時，如何有效率的搜尋出最適合的答案就變得非常困難，這時利用格羅佛量子演算法可以有效的找出最佳答案。安排選址過程中必須有許多限制條件，例如考量新的店家位置不能與同種的現存店家相鄰，進而找出的選址所在。一旦城市系統龐大，而便利商店數目很多時，格羅佛量子演算法的優勢就會顯現出來。

使用格羅佛量子演算法時，必須有效的設計演算法核心的部分，被稱爲天諭的一序列量子電路來找出最佳答案，而天諭是以平行化方式同時考慮各種量子態的可能性，並利用一連串放大量子機率幅的步驟來提高正確答案的機率。爲了使讀者能有更完整理解，下面以直觀而完整的方式介紹有關格羅佛量子演算法 [12] 的流程。

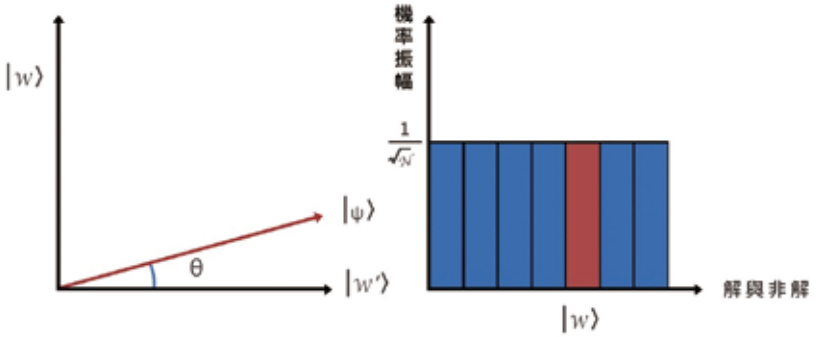

圖 8：格羅佛量子演算法步驟 1 示意圖，在各種組成態中標記出正確解，如本圖中紅色條。

步驟 1：如圖 8 右，開始計算時，要同時考量所有的可能性，先用一系列的 H 閘作用在量子線路上來準備起始之疊加態 $|\psi\rangle$。如圖 8 左圖中 $|w\rangle$ 代表正確解的部分，而 $|w'\rangle$ 則代表非解的部分，對任意起始態 $|\psi\rangle$ 中，是由正確解 $|w\rangle$ 與非解 $|w'\rangle$ 兩部分共同組成，因此在左圖中會有個 θ 角偏離非解軸 $|w'\rangle$。右圖紅色條代表任意起始態中正確答案量子態的機率幅，與其他狀態的機率完全無法分辨。接下來就是量子程式設計師的重要工作，需要設計出有關特定問題的量子電路序列－天諭，並標記出正確答案。

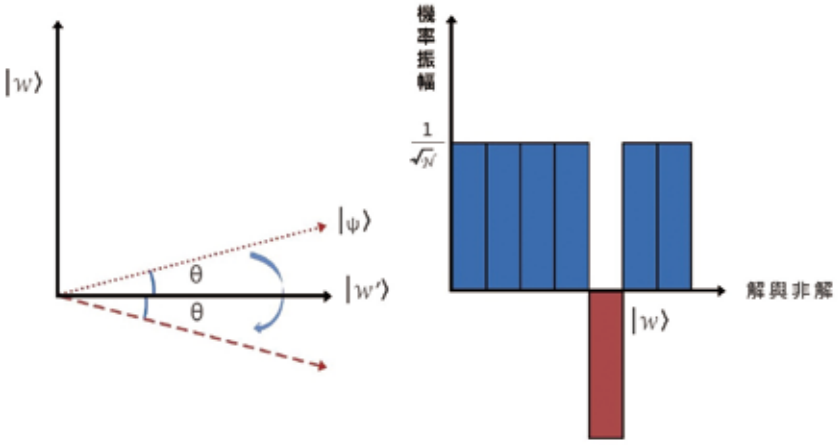

圖 9：格羅佛量子演算法步驟 2 示意圖，將標誌出的正確解的振幅予以反向操作。

步驟 2：圖 9 中，當右邊正確答案量子態的機率幅反向時，左邊圖像上 $|\psi\rangle$ 向量會沿著 $|w'\rangle$ 軸做鏡射變換 [13]。也因此造成振幅有正負的差異，會顯現出與其他狀態的不同，然而由於量子量測是機率幅的平方，所以測量結果此時沒有不同。

步驟 3：圖 10 的 $|w'\rangle$ 軸下方的虛線表達的量子態就是圖 9 中圖右邊標明的紅色條鏡射結果，如果將軸下方的虛線射線再以 $|\psi\rangle$ 爲鏡像軸，重新再做一次鏡射回去時，此時正確答案量子態的機率幅的就像圖右方的紅色的標記部份被顯著放大了。只要重複幾次這樣步驟直到正確答案可以明顯被判定，有時甚至與正確解答 $|w\rangle$ 可以幾乎完全吻合，這也就是格羅佛量子演算法給出的最後近似解 $|\psi\rangle$。

[12] **註**：參見 https://en.wikipedia.org/wiki/Grover's_algorithm。
[13] **註**：平面上基本的線性變換：旋轉、鏡射、伸縮、推移，參見 https://highscope.ch.ntu.edu.tw/wordpress/?p=51374。

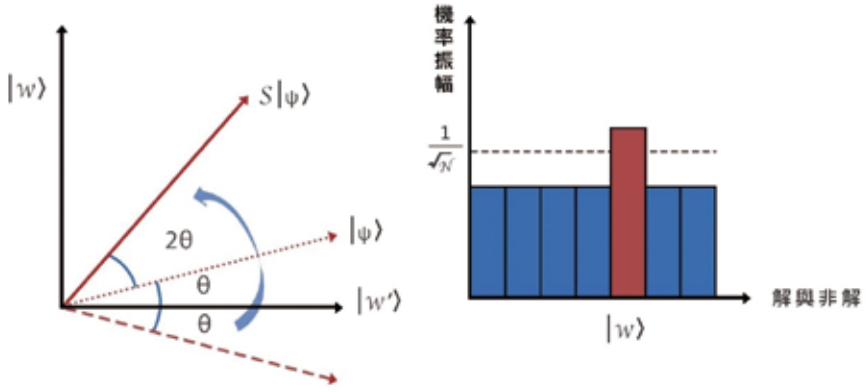

圖 10：格羅佛量子演算法步驟 3 示意圖。將圖 9 右邊的標註反向正確解，也就是本圖左邊的 $|w'\rangle$ 軸下方的虛線表達的量子態，再以 $|\psi\rangle$ 為鏡像軸做一次鏡射回去時，此時正確答案量子態近似解 $|\psi\rangle$ 的機率幅的就像圖右紅色的標記部份被顯著放大了。

格羅佛量子演算法與古典搜尋的最大差別在搜尋過程，古典演算法的搜尋是基於嘗試錯誤（try and error）的法則，也因此搜尋速度只能是 $O(n)$ 的複雜度。量子演算法的搜尋是完全量子邏輯的，由圖 8 至 10 的過程可以發現，量子搜尋的主要是標示答案在鏡射變換操作後，讓答案自動放大顯現出來，這個與古典嘗試錯誤的邏輯完全不同，也因此可以大量減少搜尋時間。

目前的量子演算法的發展正方興未艾，以上介紹只是最出名的兩種量子演算法。有興趣的讀者可以參考下表自行尋找相關資料學習 [14]，另外在演算法方面量子電腦和古典電腦的對比上，有大量計算複雜性理論的研究，有興趣讀者可以找專書研究。

[14] 註：參見 https://en.wikipedia.org/wiki/Quantum_algorithm。

核心演算法	演算法名稱	應用	對比傳統演算法的優勢	潛在可應用課題
量子傅立葉轉換	Shor's 演算法	破解RSA加密	指數型加速	密碼學 相位估計 亞群搜尋 機械學習
	HHL	求解反矩陣	指數型加速	
Grover's 算符	Grover's 演算法	搜尋問題	根號加速	未排序搜尋問題
古典量子演算混和計算	VQE	本徵值求解	在化學分子有指數型加速	新化學分子 藥物開發 金融問題 最佳化問題
	QAOA	最大割問題(max cut)		
量子退火	QAA	本徵值求解	可有效求解大矩陣的最小本徵值	機械學習 物流問題 金融問題 最佳化問題

表 1：部分著名的量子演算法及可能應用範圍。

第三節：如何開始量子計算

相信許多不少讀者看到這裏，都有躍躍欲試量子計算的衝動，下面介紹的這些量子工具會有適當幫助。量子專用軟體框架和程式語言讓研究人員可以模擬、執行和設計量子電路。2020 年的一篇評論（B. Heim et al.,"Quantum programming languages", *Nature Rev. Phys.* 2 (2020), 709~722 [15]）中描述了目前的幾種量子語言。量子程式設計語言用於管理量子硬體設備，預估量子演算法在運行時的執行成本，操作量子位元、疊加、糾纏，量測和驗證量子演算法的結果。目前量子程式設計語言主要有：

一 指令式（Imperative Programming）：指令式程式設計語言是由逐步指令組成。古典電腦中的指令式語言包括 C、JavaScript、Pascal、Python 等。現在流行的量子指令式語言有 QCL，QMASM 與 Silq。

二 函式（Functional Programming）：函式程式設計語言不用逐步指令，而是使用數學函數，也就是可以使用數學變換將輸入轉換爲輸出。函式語言不如指令式語言流行，因爲沒有迴圈指令或條件陳述式（例如 if/else 語句）。主要有 QML、Quantum Lambda Calculus、QFC 和 QPL。

三 多範式（Multi-Paradigm Programming Language）：多範式程式語言可以支援超過一種編程方式的程式語言，例如 Microsoft 的 Q# 和 Xanadu AI 的 Strawberry Fields。

微軟、IBM 和 Google 都創建了使用 Python 程式語言，分別爲 Q#、Qiskit 和 Cirq，並且建立了用戶友好開發環境和豐富的資料檔來協助編碼人員入門。例如，微軟已經提供完整的量子開發工具包（QDK），其中包含代碼庫、調試器和資源估計器，可以提前檢查量子算法需要多少量子位元。Rigetti Computing 也發布 Forest 的量子軟件開發工具包，其中包括一個名爲 pyQuil 的 Python 庫。總部位於英國的劍橋量子計算公司推出了 tket，以及相關的 pytket 庫。另一種選擇是 Silq，這是一種高階編程語言，由蘇黎世瑞士聯邦理工學院（ETH）發布，標榜能更符合量子電腦運算模式，以更精簡且容易理解形式進行編程，並且使量子電腦運算效能能更有效率發揮。在過程中更會自動識別、過濾運算過程所產生無用數據，進而讓量子算式導出正確數據。

有關量子電腦網上服務平台，目前 IBM 提供免費線上的五量子位元電腦，如果想使用更多位元的機器，必須要申請加入成爲其量子網絡的使用者。微軟通過 Azure Quantum 平台提供線上量子計算服務，谷歌目前沒有提供線上量子計算服務，亞馬遜網絡服務（AWS）是一個借用其他公司量子設備的綜合雲計算平台。目前雲端量子計算服務（Quantum Computing as-a-Servic，QCaaS） 正逐漸開始商業化服務中。對初學者來說，IBM 的 Quantum Experience 工具構建的電路是好的入門課程。邏輯閘是計算的基本線路，排列在一起的量子電路可以解決問題，將資料輸入透過天諭轉化成想要的輸出結果。但與二進制位不同，量子位元是 1

[15] 註：參見 https://www.nature.com/articles/s42254-020-00245-7。

區域	公司	所在地	量子位元種類	軟件名稱	語言	工作溫度
美洲	IBM	紐約	超導	Qiskit	Python	~10mK
	Google	加州	超導	Open Fermion Cirq	Python	~10mK
	Honeywell	美國	離子阱	Penny Lane AI	Python	−441 degree (F)
	Ion Q	美國	離子阱	Qiskit/Cirq	Python	Room Temperature
	Coldquanta	美國	冷原子	Qiskit	Python	Room Temperature
	Xanadu	美國	光子	Penny Lane	Python	Room Temperature
歐洲	OQC	英國	超導	Deltaflow/Riverlane	Python	~10mK
	Qutech	荷蘭	超導+半導	Quantum inspire	Python	~10mK
	AQT	澳洲	離子阱	Cirq/Qiskit/Pennylane	Python	RT
亞洲	本源量子	中國安徽	超導+半導	QPanda	C++/Python	~10mK
	阿里巴巴	中國杭州	超導	Alibaba Cloud Quantum Development Platform (ACQDP)	Python	~10mK
	量旋科技	中國深圳	NMR+超導	Qiskit	Python	RT
	華爲	中國深圳	超導	Hiq	C++/Python	

表 2：目前雲端之量子計算資源提供者與量子電腦種類。

和 0 的疊加，閘改變的是量子位元狀態，只有在測量時才會得到古典的數位結果。量子計算還利用了糾纏等特性，改變一個量子位元的狀態也會改變另外量子位元的狀態。這些特性使量子電腦能夠比古典電腦更快地解決某些特定的問題，例如，化學家可以使用量子計算機通過建模來加速新催化劑的識別。

即使是當今最快的量子電腦也沒有超過 100 個量子位元，並且受到隨機錯誤的困擾，導致在大型系統中計算結果的可信度有問題。2019 年，Google 展示 54 位元的量子電腦可以在幾分鐘內解決一個古典電腦需要 10,000 年才能解決的特殊亂數問題。最近中國的祖沖之量子電腦比 Google 的更快，但量子電腦至少需要數千個物理量子位元才能眞正有效模擬化學系統，達到全面量子優勢。現在的量子電腦有點像 1980 年代後期的超級電腦時期，只是證明量子電腦有能力在未來解決眞實問題。儘管如此，量子電腦近年的進展確實是日新月異，IBM 預期到 2023 年推出超過 1,000 量子位元的電腦，很多人認爲量子電腦發展時機已經成熟，越來越多的線上量子教學課程、編程語言和模擬器紛紛出現，而且確實可以在線上直接利用量子計算解決一些問題。大部分線上教學都會介紹量子計算本質上是矩陣向量乘法，例如 Quantum Computing for the Very Curious 的演練資源[16]。IBM 也建立了一個交互式工具包來配合其 Qiskit 量子語言供大眾上線練習。眞正在做量子計算必須依據量子演算法來架構出量子電路，這些電路從左到右運行，看起來有點像五線譜上有各種不同音符。類似於構建古典電子電路的 AND、OR 和 NOT 閘，在最後的測量動作之前，量子電路中的各種量子閘操控與轉換所有量子位元。IBM 的 Quantum Experience 允許使用者拖放邏輯閘來創建量子電路，然後在量子電腦上遠程執行任務。

[16] 註：請參閱 https://quantum.country/qcvc。

除了眞實的量子電腦外，也有許多古典電腦的量子模擬器可以進行量子計算。例如微軟的 QDK 有一個內置的模擬器，可以在筆記本電腦上模擬 30 個量子位元。量子模擬器可以讓你眞正看到量子態的變化，眞實量子電腦的量子態反而無法維持太久，因爲背景熱量或磁場很容易使量子位元失去量子特性。但仍應該儘量在眞正的量子電腦上進行量子計算，以掌握量子電腦嘈雜且易錯的眞實行爲。目前有關量子演算法與量子程式編寫的探索正進行中，如何最佳化量子計算是大問題，量子計算中有很多未開發的領域，隨著研究的進步和量子設備的改進，這種令人頭疼的錯誤問題將會減少。

與古典電腦的編程一樣，量子編程也是使用高級語言後，經過編譯器轉爲量子硬體的操作程序。現在混和量子運算是將量子電腦會與古典電腦在一起協作，程式使用與開發者仍是使用高階程式語言，但要採用新的量子演算法，例如 Python 的 Qiskit，所以使用工具的學習門檻並非特別困難。IBM 建議五個學習量子編程人員的必要技能 [17]

一 程式語言與演算法：需要熟習 Python、Q# 等程式語言，了解它們在量子應用下如何操作以及各如何操作以及各種量子演算法。

二 AI、機器學習與深度學習：利用古典電腦的 AI、機器學習與深度學習的技術，結合量子演算法來加速運算過程，例如變分量子演算法就是一種利用古典機器學習來優化量子變分的混和方式。

三 開源碼：以 Qiskit 爲例，它是 Python 的量子軟體工具包，建立在開源平台上，使用者需要了解各種開源平台上的功能，例如部署開源包，開發與設計等。

四 科學運算的理解：了解問題相關的特定知識與技能，才有能力眞正開發量子運算的程式碼的核心技術。

五 邏輯能力與合作：量子運算所處理的問題常是跨領域的複雜問題，解決問題需要有不同專長的技能在同一團隊彼此合作。

第四節：結語與思考題

自從量子計算出現以來，如何有效處理數據的思維方式也發生了重大變化，主要是量子現象造成程式撰寫邏輯的轉變：量子疊加與糾纏，允許數據點可以同時有多個可能值，各數據點間更彼此有特殊關聯；而量子干涉則使得數據點相互影響，甚至可以消除掉不必要的計算步驟。通過演算法設計者的巧妙技巧，量子數據間的相互影響與糾纏性可以節省大量的計算的時間和空間。多伊奇演算法的出現讓大家覺得量子演算法的優勢是不容否認的，同時 1 與 0 的數位世界將會解構後演變出奇怪的量子演算邏輯，進一步吸引專家的興趣去研發出更複雜的量子演算法，例如秀爾演算法的出現。但是什麼使得秀爾演算法如此有效？顯然糾纏與量子傅立葉變換很重要，但這不表示量子電腦在所有計算問題上都可以快過古典電腦。目前量子演算法剛開始發展，目前對各種演算法，那些適用於量子電腦，那

[17] 註：請參閱 Top Five Skills For Quantum Coding That Will Be In Demand, https://www.indiatimes.com/technology/news/quantum-coding-top-5-skills-india-jobs-ibm-548748.html。

些更適合古典電腦，或是需要量子與古典電腦的混合系統，仍有待深入研究。只有眞正適合用量子演算法的問題才能使量子電腦的運行速度超過古典電腦。一個好的量子演算法會利用量子特性將通往錯誤的答案的可能消除，而只有正確的答案被加強，這也是格羅佛演算法的精髓所在。其實這也是未來量子演算法的發展方向，把數位計算留給古典電腦，而把量子特性的計算留給量子電腦，將量子演算法與古典演算法結合起來的混合演算法是未來的方向。

不要期待量子電腦有超人力量可以瞬間解決世上所有問題，量子電腦仍然有其局限性。雖然某些現存古典密碼將來會被量子電腦快速破解，但是也會有新的抗量子密碼被發展出來。這種既競爭又合作的矛與盾的互補性才是科技進步的驅動力，也激發發展量子計算的企圖心。有了完美的矛出現，完美的盾也會隨之不久就會產生，這也是自然演化的基礎法則。

量子演算法確實可以解決許多古典演算法所無法處理的問題，數位的古典電腦即使再快速，也只是 1 與 0 交替式排列，遠不及量子電腦可以利用希爾伯特空間的特性，進而展現量子力學才是宇宙的根本之學，使用矩陣數學在多維空間內模擬宇宙的行爲，奏出熱鬧非凡，簫鼓沸天的新量子篇章，有詩爲證曰：

平盤算卦聲聲撞，
籌子多維粒粒敲，
量力自然成萬物，
鑄成矩陣沸天簫。

思考題：

1. 爲什麼我們需要設計量子演算法？
2. 多伊奇演算法爲什麼可以一石兩鳥？
3. 量子 — 古典式混合演算法的整體架構是什麼？
4. 秀爾演算法可以破解那一類加密系統？
5. 秀爾演算法可以分成哪兩個主要步驟？
6. 你能概要的說明出格羅佛量子演算法的步驟嗎？
7. 你準備如何開始量子計算，寫出你的第一個計畫並具體實施。 ∞

延伸閱讀

▶淺談量子資訊與量子計算（台灣物理協會）文 / 管希聖
https://www.ps-taiwan.org/bimonth2/download.php?d=1&cpid=165&did=1
▶量子運算的歷程和背景
https://docs.microsoft.com/zh-tw/azure/quantum/concepts-overview
▶量子世代下的密碼學：機會與挑戰
https://iis.sinica.edu.tw/zh/page/report/8106.html
▶Cleve, R., et al., “Quantum Algorithms Revisited.” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1969, The Royal Society, Jan. 1998, pp. 339~354. Crossref, doi:10.1098/rspa.1998.0164.
▶Chien-Hung Cho, Chih-Yu Chen, Kuo-Chin Chen, Tsung-Wei Huang, Ming-Chien Hsu, Ning-Ping Cao, Bei Zeng, Seng-Ghee Tan, and Ching-Ray Chang, “Quantum computation: Algorithms and Applications”, *Chinese Journal of Physics*, Volume 72, 248~269, August 2021.