UNIVERSIDAD NACIONAL DE ROSARIO

FACULTAD DE CIENCIAS EXACTAS,

INGENIERÍA Y AGRIMENSURA

TESIS DE LICENCIATURA

EN CIENCIAS DE LA COMPUTACIÓN

Agregando Medición al Cálculo de van Tonder

Autor: Alejandro Díaz-Caro

Director: Dr. Manuel Gadella

Co-Director: Dr. Pablo Martínez López

21 de Diciembre de 2007

A mi compañera
y futura esposa.

Y a mi madre.

Agradecimientos

Creo que la parte más importante de toda carrera es la de los agradecimientos. Existe un sinnúmero de gente con la que conviví, interactué, debatí, compartí y aprendí, que son los responsables de que hoy culmine mis estudios de grado. Sin ellos, nunca podría haber llegado hasta aquí.

En primer lugar quiero agradecer a mi madre, la persona que me estimuló a interesarme por la ciencia desde muy temprana edad, quien me compraba los fascículos de "Hágalo usted mismo" y "Muy interesante" en mi niñez y quien siempre estuvo allí, en el lugar indicado, para brindarme todo el apoyo emocional y estímulo que necesité a lo largo de mi carrera y de mi vida. Sin ella, nada podría haber hecho.

También debo agradecer a mis hermanas, cuñados y sobrinos: Paty, Lacho, Sol, Diego, Belén, Estefy, Marquitos y Joaquín, quienes han sabido brindarme todo su cariño, aún en mis prolongadas ausencias.

No puedo olvidar tampoco a mis tíos Fredy y Magdalena, quienes me motivaron a escribir mi primer paper y quienes a través de su ejemplo me enseñaron a valorar la ciencia.

¿Y qué sería de un estudiante si no fuese por sus compañeros y amigos? Desde el inicio de mi carrera conocí gente con la que he compartido los más gratos momentos de mi vida, desde jornadas de estudio hasta fiestas, asados y viajes. ¡Incluso más de una despedida de soltero y subsiguiente boda! A todos ellos les estoy eternamente agradecido, sin su amistad no hubiera durado ni un mes en esta ciudad. A Lucky, Cristian, Nacho, Pepi, Joselo, L0kit0, Dude, Dani, Endo, Ranto, Franco, Julián G., Julián S.F., Juanito, Brian, Martín, Lucho, Duilio, Dante, Gula, Mauro, Silvana, Valeria, Mauricio, Spe, Pablo P., Pablo B. y tantos otros que no cabrían en esta hoja... a todos ustedes: Gracias!

También quiero agradecer haber conocido a profesores tan fuera de serie como los que tiene esta facultad: Guido, su ejemplo de vida, su manera de estimular constantemente, su humildad, son algunas de sus características que nunca olvidaré. Gabriela, su interés y entusiasmo por la enseñanza, su compromiso con la ciencia, su calidez humana, su forma de ser excepcional, hacen que agradezca día a día el haberla conocido y tener el tremendo honor de ser su ayudante de cátedra. Raúl, nunca antes había conocido a una persona con tanta entrega hacia una causa: La Licenciatura en Ciencias de la Computación de la UNR. Su forma de ser y de actuar son los que hacen que esta carrera tenga ese sabor a *única en su tipo*. Además, debo agradecer a tantos otros docentes y no docentes, quienes con su entrega y dedicación estimulan y alientan día a día a tantos de nosotros: Silvia R., Nidia, Claudia, Cristina, Ana, Silvia B., Graciela, Omar, Belu y muchos más. Y quienes ya no están entre nosotros: Luchi y Peter.

A las dos personas que apostaron por mi, mis directores: Manolo y Fidel. A Fidel le estoy agradecido no sólo por su asesoramiento para este trabajo, sino también por los muchos años

de amistad y su forma de entender la docencia, que supo transmitir a través del ejemplo. A Manolo sólo puedo decir que gracias a él entré al mundo de la computación cuántica, él es el modelo de investigador que algún día quiero ser, con su humildad, dedicación, confianza y amistad he aprendido a seguir sus consejos con la confianza que se tiene por un padre bueno, que aconseja a sus hijos augurándoles lo mejor para sus vidas.

Por último, la persona que ha estado siempre a mi lado, quien siempre me apoyó y comprendió: mi futura esposa, Nache. Sólo con una persona como ella podría proyectar un futuro, alguien que siempre ha sabido comprenderme y darme la libertad de elegir un mañana, sabiendo que siempre estará a mi lado. Quien me ha hecho *tocar tierra* desde su ciencia, la social, siendo mi complemento perfecto. Quien me ha mostrado un mundo nuevo que no conocía. A ella le dedico esta tesis.

Resumen

El área de los lenguajes de programación cuánticos se ha venido desarrollando a una gran velocidad[15, 23]. En particular, se han definido varias extensiones al Lambda Cálculo que proveen la sintaxis y semántica necesarias para modelar algoritmos cuánticos.

Uno de los trabajos más influyentes en este sentido es el λ_q de André van Tonder[28]. Este es un Lambda Cálculo, definido mediante su semántica operacional, para cómputos puramente cuánticos: la medición no es parte del cálculo.

La intención del presente trabajo es agregar medición al λ_q, para lo cual se recurre a algunas herramientas del Lambda Cálculo probabilístico definido por Di Pierro, Hanking y Wiklicky en [12]. Además, siguiendo la línea de trabajo de van Tonder, se conserva la sintaxis para la lógica lineal de Philip Wadler[30].

En el Capítulo 1 se introduce un breve estado del arte, planteando los objetivos del presente trabajo.

En el Capítulo 2 se presentan los conceptos básicos de la Computación Cuántica y ejemplos necesarios para la comprensión de este trabajo.

En el Capítulo 3 se desarrolla el modelo de Cálculo λ_q de van Tonder y se presenta una breve discusión sobre las implicancias del hecho de que la medición cuántica no sea parte del cálculo. Para ello, se recurre al ejemplo del algoritmo de Teleportación[9], en el cual se debe diferir la medición.

En el Capítulo 4 se desarrollan las modificaciones necesarias al λ_q para que éste incluya la medición, y se vuelve a introducir el ejemplo de Teleportación a fin de resaltar la utilidad de los cambios propuestos.

Por último, en el Capítulo 5 se introduce una breve discusión sobre los resultados obtenidos y se plantean algunos interrogantes abiertos para trabajos futuros.

Índice general

Capítulo 1

Introducción

1.1. Antecedentes y estado del arte

Se podría considerar que el área de los lenguajes de programación cuánticos empezó en 1996, con los aportes de Gregory Baker[7], quien desarrolló un lenguaje imperativo de simulación cuántica, y Emanuel Knill[18], quien brindó los lineamientos para un pseudocódigo cuántico. En el mismo año, pero desde un punto de vista funcional, Philip Maymin[20] realizó la primera extensión al λ-cálculo para incluir fenómenos cuánticos. Si bien los lenguajes mencionados no son completos, contienen las ideas esenciales utilizadas en desarrollos posteriores, como el uso de la lógica lineal.

A partir de 2001 comienzan a producirse en esta área gran cantidad de trabajos, sobre todo, desde el paradigma funcional[28, 24, 5, 27, 2, 6, 3, 25, 16, 4, 29].

Uno de los trabajos más influyentes desde el punto de vista del λ-cálculo es el de André van Tonder[28, 29], el cual, al ser no tipado, permanece simple. Si bien su desarrollo es muy completo, no incluye la operación medición en el cálculo (ver discusión en Sección 3.8).

Benoît Valiron[27], en un principio, y luego junto a Peter Selinger[25], agregaron medición a este cálculo definiendo un lenguaje de programación funcional cuántico, basándose en una idea previa de Selinger[24] de tener un control clásico y datos cuánticos.

Por otro lado, Pablo Arrighi y Gilles Dowek[5, 6] también definieron un λ-cálculo, pero basado en álgebra lineal, y realizaron una discusión sobre la relación existente entre el álgebra lineal y la lógica lineal.

Thorsten Altenkirch y Jonathan Grattage, siguiendo la línea de Selinger[24], definieron el QML[2, 3, 16], un lenguaje funcional pero con control y datos cuánticos. Además, junto a Juliana Vizzotto y Amr Sabry[4] desarrollaron una teoría ecuacional que, tal como lo hizo previamente van Tonder[28], no incluye la operación medición.

Para una revisión más exhaustiva acerca de los distintos desarrollos de lenguajes de programación, tanto funcionales como imperativos, se recomienda la lectura de [15] y [23].

1.2. Descripción de los objetivos

El objetivo principal de este trabajo es dotar al λ-cálculo de André van Tonder de la operación medición. A diferencia de los trabajos de Selinger y Valiron, la idea es hacerlo mediante un λ-cálculo lineal puramente cuántico, con reglas de transición probabilísticas y no tipado, manteniendo así la simplicidad del cálculo.

Capítulo 2

Computación Cuántica

2.1. Introducción

La Computación Cuántica, desde un punto de vista algorítmico, plantea un nuevo modelo de Computación. Este modelo incorpora nuevas formas de *pensar* los algoritmos. Conceptos claves para ello son el enredo cuántico, el paralelismo, la medición y el teorema de no-cloning, los cuales no tienen analogía con ningún concepto cásico.

Esta rama de las Ciencias de la Computación tiene su origen en la física, y más precisamente en el físico estadounidense Richard Feynman, quien planteó la crucial pregunta *¿Podríamos construir una nueva clase de computadora que imite cualquier sistema cuántico del mundo real?*[10]. Este interrogante, lejos de ser una solución, abre las puertas de conceptos nunca antes concebidos. ¿Qué ganancia se lograría si las computadoras fuesen regidas por las leyes de la mecánica cuántica?.

Simplificando un poco los conceptos, podemos pensar a una computadora cuántica como una máquina para automatizar experimentos de la mecánica cuántica. De esta manera no estaríamos *simulando* la realidad, sino que tendríamos una máquina de propósito general que lleve a cabo los experimentos.

Volviendo a las Ciencias de la Computación, fueron el algoritmo de Lov Grover[17] y el algoritmo de Peter Shor[26] los cuales despertaron el gran interés desde esta ciencia por la computación cuántica. El primero es un algoritmo de búsqueda sobre registros desordenados, el cual provee una ganancia cuadrática de complejidad temporal frente a cualquier algoritmo conocido. El segundo es un algoritmo para factorizar números, con una ganancia exponencial. Este último generó el interés de los departamentos de defensa de las grandes potencias mundiales, ya que contar con una computadora cuántica que pueda llevar a cabo dicha hazaña en un corto tiempo implicaría la ruptura de los sistemas criptográficos más seguros que existen en la actualidad. Esto causó una gran inversión monetaria, pública y privada, hacia la investigación en el área.

En las siguientes secciones brindamos las definiciones matemáticas de este modelo compu-

tacional, y algunos ejemplos de algoritmos que ayudan a entender los principales conceptos.

2.2. Conceptos previos

En esta sección damos los conceptos matemáticos y la notación necesarios para el resto del capítulo.

2.2.1. Espacio de Hilbert

Definición 2.1 *Sea E un espacio lineal sobre $\mathbb{K}$. Un* producto interno definido sobre E *es una aplicación $\langle , \rangle : E \times E \to \mathbb{K}$ que verifica ser:*

- *Definida positiva:*
  $\langle x, x\rangle \geq 0, \forall x \in E \ \wedge \ \langle x, x\rangle = 0 \Leftrightarrow x = 0_E$.
- *Lineal por derecha:*
  $\langle z, \lambda x + \mu y\rangle = \lambda\langle z, x\rangle + \mu\langle z, y\rangle, \forall x, y, z \in E, \forall \lambda, \mu \in \mathbb{K}$.
- *Hermítica:*
  $\langle x, y\rangle = \overline{\langle y, x\rangle}, \forall x, y \in E$.

Nota 2.2 *Una consecuencia directa de la Linealidad por derecha y Hermeticidad es la Antilinealidad por izquierda:*
$\langle \lambda x + \mu y, z\rangle = \overline{\lambda}\langle x, z\rangle + \overline{\mu}\langle y, z\rangle, \forall x, y, z \in E, \forall \lambda, \mu \in \mathbb{K}$.

Definición 2.3 *Un espacio lineal sobre $\mathbb{K}$ con producto interno es un espacio* pre-Hilbert

Nota 2.4 *Todo espacio pre-Hilbert es un espacio lineal normado, con la norma $\|x\| = \sqrt{\langle x, x\rangle}$.*

Definición 2.5 *Sea X_n una sucesión de vectores de un espacio V.*

Si $\|X_k - X_m\| \to 0$ cuando $k, m \to \infty$, entonces la sucesión X_n es una sucesión de Cauchy*.*

O lo que es lo mismo: si $\forall \varepsilon > 0, \exists N \in \mathbb{N} \not/$ si $k, m \geq N, \|X_k - X_m\| < \varepsilon$ entonces la sucesión X_n es una sucesión de Cauchy.

Nota 2.6 *Observaciones:*

- *Esto quiere decir que se puede hacer distar entre sí los términos tan poco como se quiera.*
- *Toda sucesión convergente es de Cauchy (pero no a la inversa).*

Ejemplo 2.7 *Una sucesión de Cauchy no convergente:*

Sea F el espacio vectorial de funciones reales contínuas en $[0,1]$ con producto interno definido como:

$$\langle f, g\rangle = \int_0^1 f(x)g(x)dx,$$

y sea la sucesión $\{f_n\}$ con

$$f_n(x) = \begin{cases} 1 & si\ 0 \leq x \leq \frac{1}{2} \\ 1-(x-\frac{1}{2})n & si\ \frac{1}{2} < x < \frac{1}{2}+\frac{1}{n} \\ 0 & si\ \frac{1}{2}+\frac{1}{n} \leq x \leq 1 \end{cases}.$$

$\{f_n\}$ es una sucesión de Cauchy, ya que

$$\|f_n - f_m\|^2 = \langle f_n - f_m, f_n - f_m\rangle =$$

$$= \int_0^1 |\ (f_n - f_m)(x)\ |^2 dx = \int_{\frac{1}{2}}^{\frac{1}{2}+max\{\frac{1}{n},\frac{1}{m}\}} |\ (f_n - f_m)(x)\ |^2 dx \leq \varepsilon,$$

pero $\{f_n\}$ no converge, ya que cuando $n \to \infty$, esta sucesión tiende a una función discontinua (y el espacio F es el espacio de funciones reales continuas en $[0,1]$).

Definición 2.8 *V es* completo *para la norma $\|\cdot\|$, si y sólo si toda sucesión de Cauchy converge.*

Definición 2.9 *Un espacio pre-Hilbert completo en su norma se denomina* espacio de Hilbert*.*

2.2.2. Productos tensoriales

Definición 2.10 *El producto* tensorial de dos matrices*, P de orden $n \times m$ y Q de orden $k \times l$, se define como la matriz*

$$P \otimes Q = \begin{pmatrix} p_{11}Q & \dots & p_{1m}Q \\ \vdots & & \vdots \\ p_{n1}Q & \dots & p_{nm}Q \end{pmatrix}.$$

Nota 2.11 *En particular, tomando las matrices P de orden $n \times 1$ y Q de orden $k \times 1$, se obtiene el* producto tensorial entre vectores*.*

Ejemplo 2.12

$$P \otimes Q = \begin{pmatrix} p_{11} \begin{pmatrix} q_{11} \\ \vdots \\ q_{1k} \end{pmatrix} \\ \vdots \\ p_{1m} \begin{pmatrix} q_{11} \\ \vdots \\ q_{1k} \end{pmatrix} \end{pmatrix}.$$

Definición 2.13 *El* producto tensorial de espacios vectoriales*, E y F, se define como sigue.*

Sea $B_1 = \{e_i\}_{i=1,\ldots,dim(E)}$ una base de E, y $B_2 = \{f_j\}_{j=1,\ldots,dim(F)}$ una base de F. Entonces, el conjunto $\{e_i \otimes f_j\}_{i=1,\ldots,dim(E),j=1,\ldots,dim(F)}$ es una nueva base, y el espacio generado por ella es el espacio $E \otimes F$.

En símbolos: $\mathfrak{L}(B_1 \otimes B_2) = E \otimes F$.

Ejemplo 2.14 *Sea $B_1 = \{v_1, v_2\}$ y $B_2 = \{u_1, u_1\}$, entonces*
$B_1 \otimes B_2 = \{v_1 \otimes u_1, v_1 \otimes u_2, v_2 \otimes u_1, v_2 \otimes u_2\}$.

Proposición 2.15 *Existen vectores de $E \otimes F$ que no son producto tensorial de un vector de E y uno de F.*
Demostración: *Sea*

$$v = \begin{pmatrix} \alpha \\ 0 \\ 0 \\ \beta \end{pmatrix} \text{ con } \alpha, \beta \neq 0.$$

Supongamos que existen dos vectores, tales que el producto tensorial es igual a v, entonces:

$$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix} = \begin{pmatrix} \alpha \\ 0 \\ 0 \\ \beta \end{pmatrix} \Rightarrow \begin{cases} ac = \alpha \\ ad = 0 \\ bc = 0 \\ bd = \beta \end{cases}$$

y este sistema no tiene solución. □

2.2.3. Notación de Dirac

Esta notación, introducida por Paul Dirac[14], nos permite identificar los vectores de una manera concisa.

Llamamos *ket* a un vector columna ψ y lo notamos con $|\psi\rangle$, y llamamos *bra* a su transpuesto conjugado y lo notamos por $\langle\psi|$.

Ejemplo 2.16 *Sean*

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad y \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Entonces, se puede considerar las combinaciones lineales de $|0\rangle$ *y* $|1\rangle$ *de la siguiente manera:*

$$\alpha\,|0\rangle + \beta\,|1\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}.$$

En general, se pueden definir cosas como

$$|+\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad \text{y} \quad |-\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix},$$

y dado que estos son dos vectores ortogonales (por ende, forman base), también se puede escribir cualquier vector como combinación lineal de $|+\rangle$ y $|-\rangle$.

Ejemplo 2.17

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha\,|0\rangle + \beta\,|1\rangle = \frac{1}{\sqrt{2}}(\alpha+\beta)\,|+\rangle + \frac{1}{\sqrt{2}}(\alpha-\beta)\,|-\rangle\,.$$

Nota 2.18 *Se considerará de aquí en más, excepto que se indique lo contrario, el espacio complejo de dimensión* N, $\mathbb{C}^N$.

Nota 2.19 *La notación admite diferentes maneras de representar un mismo ket. Por ejemplo* $|\lambda_1\psi_1 + \lambda_2\psi_2\rangle$ *se puede escribir también como* $\lambda_1\,|\psi_1\rangle + \lambda_2\,|\psi_2\rangle$.

Definición 2.20 *Sean*

$$|\psi\rangle = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} \; y \; |\phi\rangle = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_N \end{pmatrix}.$$

Llamamos braket *a la siguiente operación*

$$\langle\psi|\phi\rangle = (\alpha_1^*, \ldots, \alpha_N^*) \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_N \end{pmatrix} = a \in \mathbb{C}.$$

Proposición 2.21 *La operación braket define un producto interno en el espacio de Hilbert* $\mathbb{C}^N$.
Demostración: *Definida positiva:*

$$\langle\psi|\psi\rangle = \sum_{i=1}^{N} |\alpha_i|^2 \geq 0.$$

Lineal por derecha:

$$\langle\psi|\lambda_1\phi_1 + \lambda_2\phi_2\rangle = \lambda_1\langle\psi|\phi_1\rangle + \lambda_2\langle\psi|\phi_2\rangle.$$

Antilineal por izquierda:

$$\langle\lambda_1\psi_1 + \lambda_2\psi_2|\phi\rangle = \lambda_1^*\langle\psi_1|\phi\rangle + \lambda_2^*\langle\psi_2|\phi\rangle.$$

Hermítica:

$$\langle\psi|\phi\rangle = \langle\phi|\psi\rangle^*.$$

□

Definición 2.22 *Dado un conjunto* $B = \{|u_i\rangle\}_N$*, se dice que* B *es una base ortonormal de* $\mathbb{C}^N$ *sii*

$$\langle u_i|u_j\rangle = \delta_{ij}$$

y

$$\forall x \in \mathbb{C}^N;\ \langle x|u_i\rangle = 0,\ i = 1, 2, \ldots, N \Rightarrow x = \mathbf{0}.$$

De esta definición se concluye que todo Ket $|\psi\rangle$ se puede expresar como

$$|\psi\rangle = \sum_{i=1}^{N} a_i\,|u_i\rangle,$$

donde $a_i = \langle u_i|\psi\rangle \in \mathbb{C}$, ya que

$$\langle u_i|\psi\rangle = \langle u_i|\sum_{j=1}^{N} a_j\,|u_j\rangle = \sum_{j=1}^{N} a_j \underbrace{\langle u_i|u_j\rangle}_{\delta_{ij}} = a_i.$$

Definición 2.23 *Toda base ortonormal* $B = \{|u_i\rangle\}_N$ *cumple con la siguiente* condición de clausura*:*

$$\sum_{i=1}^{N} |u_i\rangle\,\langle u_i| = I,$$

ya que

$$\left(\sum_{i=1}^{N} |u_i\rangle \langle u_i|\right) |\psi\rangle = \left(\sum_{i=1}^{N} |u_i\rangle \langle u_i|\right) \left(\sum_{j=1}^{N} a_j |u_i\rangle\right)$$

$$= \sum_{i,j=1}^{N} a_j |u_i\rangle \underbrace{\langle u_i|u_j\rangle}_{\delta_{ij}} = \sum_{i=1}^{N} a_i |u_i\rangle = |\psi\rangle .$$

Además, a todo Bra $\langle\phi|$ lo podemos escribir como

$$\langle\phi| = \sum_{i=1}^{N} b_i^* \langle u_i|,$$

donde $b_i^* = \langle\phi|u_i\rangle \in \mathbb{C}$, ya que

$$\langle\phi| = \langle\phi| \underbrace{\left[\sum_{i=1}^{N} |u_i\rangle \langle u_i|\right]}_{I} = \sum_{i=1}^{N} \langle\phi|u_i\rangle \langle u_i| \Rightarrow b_i^* = \langle\phi|u_i\rangle.$$

De aquí en más, nos referiremos sólo a los vectores normalizados (con norma 1) de $\mathbb{C}^N$. Esto es

$$1 = \|\psi\|^2 = \langle\psi|\psi\rangle = \left(\sum_{j=1}^{N} a_j^* \langle u_j|\right) \left(\sum_{i=1}^{N} a_i |u_i\rangle\right) =$$

$$\sum_{i,j=1}^{N} a_j^* a_i \underbrace{\langle u_j|u_i\rangle}_{\delta_{ij}} = \sum_{i=1}^{N} |a_i|^2 = 1.$$

2.2.4. Representación de Operadores

Un operador A es una matriz de la forma

$$A = \left(\underbrace{\sum_{i=1}^{N} |u_i\rangle \langle u_i|}_{I}\right) A \left(\underbrace{\sum_{j=1}^{N} |u_j\rangle \langle u_j|}_{I}\right) =$$

$$\sum_{i,j=1}^{N} |u_i\rangle \underbrace{\langle u_i| \overbrace{A |u_j\rangle}^{|-\rangle}}_{\alpha_{ij}} \langle u_j| = \sum_{i,j=1}^{N} \alpha_{ij} |u_i\rangle \langle u_j|,$$

entonces, los elementos de la matriz de A son $(\alpha_{ij})_N$.

Veamos esto aplicando el operador A a un Ket $|\psi\rangle$ cualquiera:

$$A|\psi\rangle = \left(\sum_{i,j=1}^{N} \alpha_{ij} |u_i\rangle \langle u_j|\right)\left(\sum_{k=1}^{N} a_k |u_k\rangle\right) =$$

$$\sum_{i,j,k=1}^{N} \alpha_{ij} a_k |u_i\rangle \underbrace{\langle u_j|u_k\rangle}_{\delta_{ij}} = \sum_{i,j=1}^{N} \alpha_{ij} a_j |u_i\rangle,$$

entonces, las componentes del vector $A|\psi\rangle$, son

$$b_i = \sum_{j=1}^{N} \alpha_{ij} a_j.$$

Viendo esto en notación matricial, se tiene:

$$\begin{array}{c|c} & \begin{pmatrix} a_1 \\ \vdots \\ a_N \end{pmatrix} \\ \hline \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1N} \\ \vdots & & \vdots \\ \alpha_{N1} & \cdots & \alpha_{NN} \end{pmatrix} & \begin{pmatrix} \sum_{j=1}^{N} \alpha_{1j} a_j \\ \vdots \\ \sum_{j=1}^{N} \alpha_{Nj} a_j \end{pmatrix}. \end{array}$$

Definición 2.24 *El* adjunto *de un operador A, que se nota por $A^\dagger$, se define de la siguiente manera:*

$$\langle\phi| A |\psi\rangle^* = \langle\psi| A^\dagger |\phi\rangle.$$

Nota 2.25 *Recordando que $\alpha_{ij} = \langle u_i| A |u_j\rangle$, se deduce que las componentes de $A^\dagger$ son*

$$\alpha_{ji}^* = \langle u_j| A |u_i\rangle^* = \langle u_i| A^\dagger |u_j\rangle.$$

$\therefore A^\dagger = (A^*)^T$.

Propiedades 1 *Sean A y B operadores de $\mathbb{C}^N$, $\lambda \in \mathbb{C}$ y $|\phi\rangle \in \mathbb{C}^N$. Entonces*

- $(A^\dagger)^\dagger = A$,
- $(A+B)^\dagger = A^\dagger + B^\dagger$,
- $(\lambda A)^\dagger = \lambda^* A^\dagger$,

- $(AB)^\dagger = B^\dagger A^\dagger$ *y*
- $\langle A\phi| = \langle\phi| A^\dagger$.

Definición 2.26 *Al operador* $P \equiv |\phi\rangle \langle\phi|$ *se le llama* proyector simple*, ya que proyecta ortogonalmente un Ket* $|\psi\rangle$ *cualquiera sobre el Ket* $|\phi\rangle$.

O sea,

$$P|\psi\rangle = |\phi\rangle \underbrace{\langle\phi|\psi\rangle}_{c_j \in \mathbb{C}} = c_j |\phi\rangle .$$

Definición 2.27 *El operador A de dice* hermítico *si y sólo si* $A = A^\dagger$.

Nota 2.28 *Si un operador es hermítico, su diagonal debe ser real, ya que* $\alpha_{ij} = \alpha^*_{ji} \Rightarrow \alpha_{ii} = \alpha^*_{ii}$.

Definición 2.29 *El operador U se dice* unitario *si y sólo si* $U^\dagger U = I$*, o lo que es lo mismo, si* $A^\dagger = A^{-1}$.

Propiedades 2 *Para cualquier operador unitario U vale lo siguiente:*

- *U preserva el producto interno. Esto es*

$$\langle U\phi|U\psi\rangle = \langle\phi| \underbrace{U^\dagger U}_{I} |\psi\rangle = \langle\phi|\psi\rangle.$$

- U^{-1} *es unitario.*
- *Si* $\{|\psi_i\rangle\}_N$ *es base ortonormal, entonces* $\{U|\psi_i\rangle\}_N$ *también lo es.*

Definición 2.30 *Un conjunto de matrices* $\{M_i\}_k$ *se dice que es un* operador de medición *si satisface que*

$$\sum_{i=1}^{k} M_i M_i{}^\dagger = I.$$

Definición 2.31 *Se dice que un sistema representado por un Ket* $|\phi\rangle$ evoluciona *de dos maneras posibles:*

- *Por la aplicación de un operador unitario y hermítico U:* $|\phi\rangle \xrightarrow{U} |\psi\rangle$ *(O lo que es equivalente* $|\psi\rangle = U|\phi\rangle$*).*

- *Por la aplicación de un operador de medición de la siguiente manera:*

$$|\phi\rangle \xrightarrow{\{M_i\}_k} |\psi\rangle,$$

donde

$$|\psi\rangle = \frac{M_i |\phi\rangle}{\sqrt{\langle\phi| {M_i}^\dagger M_i |\phi\rangle}},$$

para algún $1 \leq i \leq k$.

No se puede saber qué M_i *se va a aplicar, sólo se conoce su probabilidad, que viene dada por la siguiente ley:*

$$p(i) = \langle\phi| {M_i}^\dagger M_i |\phi\rangle,$$

donde $p(i)$ *denota la probabilidad que se aplique la matriz* M_i.

Ejemplo 2.32 *Sea el siguiente operador medición:*

$$M_0 = |0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad y \quad M_1 = |1\rangle\langle 1| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Notar que $M_0{M_0}^\dagger + M_1{M_1}^\dagger = M_0 + M_1 = I$. $\therefore \{M_0, M_1\}$ *es un operador de medición. Cabe destacar que en este caso* M_0 *y* M_1 *son proyectores simples.*

Sea $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. *Entonces, la probabilidad de que al realizar la medición se aplique* M_0 *es*

$$p(0) = \langle\psi| {M_0}^\dagger M_0 |\psi\rangle = (\alpha^*\langle 0| + \beta^*|1\rangle) M_0 (\alpha|0\rangle + \beta|1\rangle)$$

$$= |\alpha|^2 \underbrace{\langle 0| \overbrace{M_0|0\rangle}^{|0\rangle}}_{1} + \alpha^*\beta \underbrace{\overbrace{\langle 0| M_0}^{\langle 0|} |1\rangle}_{0} + \alpha\beta^* \underbrace{\langle 1| \overbrace{M_0|0\rangle}^{|0\rangle}}_{0} + |\beta|^2 \underbrace{\langle 1| \overbrace{M_0|1\rangle}^{\overline{0}}}_{0} = |\alpha|^2.$$

Análogamente

$$p(1) = \langle\psi| {M_1}^\dagger M_1 |\psi\rangle = |\beta|^2.$$

Notar que, dado que el vector está normalizado, se cumple lo siguiente

$$p(0) + p(1) = |\alpha|^2 + |\beta|^2 = 1.$$

Si se aplicó M_0, *el sistema evoluciona al siguiente estado:*

$$\frac{M_0|\psi\rangle}{\sqrt{\langle\psi| {M_0}^\dagger M_0 |\psi\rangle}} = \frac{M_0|\psi\rangle}{\sqrt{p(0)}} = \frac{\alpha}{|\alpha|}|0\rangle,$$

el cual está normalizado, ya que

$$\left|\frac{\alpha}{|\alpha|}\right|^2 = \frac{|\alpha|^2}{|\alpha|^2} = 1.$$

Análogamente si se aplicó M_1, el sistema evoluciona a

$$\frac{M_1 |\psi\rangle}{\sqrt{p(1)}} = \frac{\beta}{|\beta|} |1\rangle .$$

2.3. Qubits

Definición 2.33 *Un* qubit *o bit cuántico es un vector con norma* 1 *del espacio de Hilbert* $\mathbb{C}^2$.

Considerando la base $\{|0\rangle, |1\rangle\}$ de $\mathbb{C}^2$, cualquier qubit puede escribirse como

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle ,$$

donde $|\alpha|^2 + |\beta|^2 = 1$.

Definición 2.34 *Un sistema de N-qubits es un vector con norma* 1 *del espacio* $\bigotimes_{i=1}^{N} \mathbb{C}^2$.

Nota 2.35 *La base canónica del espacio* $\bigotimes_{i=1}^{N} \mathbb{C}^2$ *es*

$$\{|0\ldots00\rangle, |0\ldots01\rangle, \ldots, |1\ldots11\rangle\} = \{|i\rangle\}_{i=0,\ldots,2^N-1}$$

.

Definición 2.36 *Un* algoritmo cuántico *consiste en la evolución de un sistema representado por N-qubits.*

Definición 2.37 *A los operadores unitarios hermíticos se los llama* compuertas cuánticas.

Las compuertas cuánticas más importantes, por su utilidad en el diseño de algoritmos, son las siguientes:

- Compuerta Hadamard o H:

$$\begin{array}{l} H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{array} \quad \text{donde:} \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} .$$

- La identidad o I:

$$\begin{array}{l} I\,|0\rangle = |0\rangle \\ I\,|1\rangle = |1\rangle \end{array} \quad \text{donde:} \quad I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

- La negación o X:

$$\begin{array}{l} X\,|0\rangle = |1\rangle \\ X\,|1\rangle = |0\rangle \end{array} \quad \text{donde:} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

- El cambio de fase o Z:

$$\begin{array}{l} Z\,|0\rangle = |0\rangle \\ Z\,|1\rangle = -\,|1\rangle \end{array} \quad \text{donde:} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

- La Controlled-Not o $CNOT$:

$$\begin{array}{l} CNOT\,|0x\rangle = |0x\rangle \\ CNOT\,|1x\rangle = |1\rangle \otimes X\,|x\rangle \end{array} \quad \text{donde:} \quad CNOT = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}.$$

En particular, a las matrices I, X, iXZ y Z se las llaman *matrices de Pauli.*

2.4. Teorema de No-Cloning

Un teorema de suma importancia en la Computación Cuántica es el Teorema de No-Cloning[31]. Se dá una explicación matemática del teorema y en la siguiente subsección se discuten sus implicaciones.

Teorema 2.38 *No existe U, operador unitario y hermítico, tal que para algún $|\varphi\rangle \in \mathbb{C}^N$ y $\forall\,|\psi\rangle \in \mathbb{C}^N$ se cumpla que*

$$U\,|\psi\varphi\rangle = |\psi\psi\rangle\,.$$

□

Antes de proceder a demostrar este teorema, se enuncia una propiedad del producto interno *braket*:

Propiedades 3

$$\langle ab|cd\rangle = \langle a|c\rangle\langle b|d\rangle \quad \forall a,b,c,d \in \mathbb{C}^N.$$

Demostración:

$$\langle ab|cd\rangle = \sum_{i=1}^{N} a_i^* c_i \sum_{j=1}^{N} b_j^* d_j = \langle a|c\rangle\langle b|d\rangle \quad \forall a,b,c,d \in \mathbb{C}^N.$$

□

Utilizando esta propiedad, se puede llevar a cabo la demostración del teorema:

Demostración del Teorema de No-Cloning

Supóngase que existe tal operación U, de la cual se habla en el teorema. Entonces, dados cualesquiera $|\psi\rangle, |\phi\rangle \in \mathbb{C}^N$, se cumple

$$U|\psi\varphi\rangle = |\psi\psi\rangle$$

$$\text{y } U|\phi\varphi\rangle = |\phi\phi\rangle .$$

Esto significa que se debe cumplir la siguiente igualdad

$$\underbrace{\langle U\psi\varphi|U\phi\varphi\rangle}_{(1)} = \underbrace{\langle\psi\psi|\phi\phi\rangle}_{(2)} .$$

Pero

$$(1) = \langle U\psi\varphi|U\phi\varphi\rangle = \langle\psi\varphi|\, U^\dagger U\,|\phi\varphi\rangle = \langle\psi\varphi|\phi\varphi\rangle = \langle\psi|\phi\rangle \underbrace{\langle\varphi|\varphi\rangle}_{1} = \langle\psi|\phi\rangle,$$

y

$$(2) = \langle\psi\psi|\phi\phi\rangle = \langle\psi\psi|\phi\phi\rangle = \langle\psi|\phi\rangle\langle\psi|\phi\rangle = \langle\psi|\phi\rangle^2.$$

$$\therefore \langle\psi|\phi\rangle = \langle\psi|\phi\rangle^2 \Rightarrow \begin{cases} \langle\psi|\phi\rangle = 0 & \text{ó} \\ \langle\psi|\phi\rangle = 1 \end{cases}$$

Este producto no puede ser 0, ya que $|\psi\rangle$ y $|\phi\rangle$ son dos Kets cualesquiera y tampoco 1, ya que eso significaría que son iguales. □

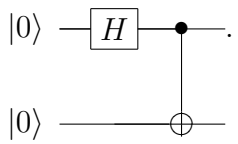
2.4.1. Implicaciones

El teorema de No-Cloning tiene una implicación muy fuerte: Asegura que no existe una máquina genérica que nos permita copiar cualquier qubit dado.

Ésto, sumado a la medición cuántica, implica que si se tiene un qubit cualquiera, luego de medirlo *se perderá definitivamente parte de la información que éste contenía*, ya que la medición cambia el estado del qubit y no existe manera de clonarlo previamente.

2.5. Circuitos cuánticos

Los *circuitos cuánticos* son representaciones gráficas de los algoritmos cuánticos. Por ejemplo, un algoritmo que recibe como entrada dos qubits, aplica una compuerta Hadamard al primero y luego un CNOT entre el primero y el segundo, se representa por



Los circuitos se leen de izquierda a derecha. Cada línea representa una conexión en el circuito. La compuerta Hadamard se ha representado con un cuadrado con una H dentro y el CNOT, tiene un punto en el qubit de control y una cruz en el target.

2.6. Enredo cuántico

Definición 2.39 *Un qubit* enredado *o* entangled *es un qubit con dim* ≥ 2*, el cual no es producto tensorial de qubits de menor dimensión.*

A los estados enredados también se les llama estados EPR debido a la paradoja planteada por Einstein, Podolsky y Rosen[13]. Dicha paradoja dice que si se tiene un par enredado, por más lejano que estén físicamente un qubit del otro, al efectuar una medición sobre uno de ellos, el otro qubit también se verá afectado.

Esto se puede apreciar mejor con un pequeño ejemplo.

Ejemplo 2.40 *Sea el operador medición* $M = \{M_0, M_1\}$*, donde*

$$M_0 = |0\rangle \langle 0|$$
$$y\ M_1 = |1\rangle \langle 1| .$$

Aplicando este operador al primer qubit del estado $\beta_{00} = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ *ocurre lo siguiente:*

Si se aplica M_0 *(el cual lo expresamos como* $M_0 \otimes I$ *para que se aplique* M_0 *al primer qubit y la identidad al segundo), el estado resultante será*

$$\frac{(M_0 \otimes I)\beta_{00}}{\sqrt{p(0)}}$$

$$= \frac{(|00\rangle\langle 00| + |01\rangle\langle 01|)\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)}{\sqrt{\frac{1}{\sqrt{2}}(\langle 00| + \langle 11|)(|00\rangle\langle 00| + |01\rangle\langle 01|)\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)}}$$

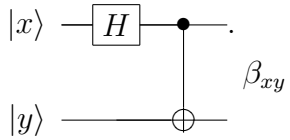
$$= \frac{\frac{1}{\sqrt{2}}(|00\rangle\langle 00|00\rangle)}{\sqrt{\frac{1}{2}\langle 00|00\rangle\langle 00|00\rangle}} = |00\rangle\,.$$

Análogamente, si se aplica M_1 al primer qubit se obtendrá $|11\rangle$.

Como se puede apreciar, si el primer qubit del par evoluciona mediante una medición, el segundo también lo hace, aún cuando no se realiza medición sobre dicho qubit.

2.6.1. Estados de Bell

Sea el siguiente circuito cuántico



Si en la entrada del circuito se ingresan qubits definidos (*i.e.* qubits pertenecientes a la base), ocurre lo siguiente:

- $|00\rangle$

$$|00\rangle \xrightarrow{H(1)} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$
$$\xrightarrow{CNOT(1,2)} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \beta_{00}.$$

- $|01\rangle$

$$|01\rangle \xrightarrow{H(1)} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|1\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |11\rangle)$$
$$\xrightarrow{CNOT(1,2)} \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) = \beta_{01}.$$

- $|10\rangle$

$$|10\rangle \xrightarrow{H(1)} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)|0\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |10\rangle)$$
$$\xrightarrow{CNOT(1,2)} \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) = \beta_{10}.$$

- $|11\rangle$

$$|11\rangle \xrightarrow{H(1)} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)|1\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |11\rangle)$$
$$\xrightarrow{CNOT(1,2)} \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) = \beta_{11}.$$

A estos cuatro estados se los llama *Estados de Bell*, los cuales son estados enredados de $\mathbb{C}^4$.

Nota 2.41 $\beta_{00} = (X \otimes I)\beta_{01} = (Z \otimes I)\beta_{10} = (XZ \otimes I)\beta_{11}$.

2.7. Teleportación Cuántica

El objetivo de este algoritmo, desarrollado por Bennett *et. al.*[9], es transmitir un qubit entre un emisor y un receptor mediante el envío de dos bits clásicos. Históricamente se llama Alice al emisor y Bob al receptor.

Los pasos a seguir por Alice y Bob son los siguientes.

Paso 1. Alice y Bob preparan un estado β_{00}.

Paso 2. Alice se queda con el primer qubit del par y Bob se lleva el segundo.

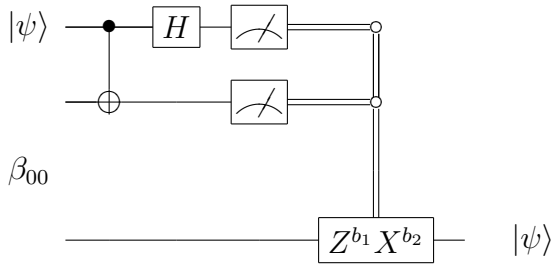
Paso 3. Alice aplica CNOT entre el qubit a transmitir y el primero del par β_{00}.

Paso 4. Alice aplica Hadamard al primero de sus dos qubits, luego realiza una medición sobre ambos y envía el resultado de la medición (los cuales pueden encodearse mediante 2 bits clásicos) a Bob.

Paso 5. Bob aplica una transformación sobre su qubit, de acuerdo a los bits recibidos, basándose en la siguiente tabla

Bits recibidos	00	01	10	11
Compuerta a aplicar	I	X	Z	ZX

El circuito completo queda de la siguiente manera



donde $|\psi\rangle$ es el qubit a teleportar.

Veamos, sea $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, entonces

$$|\psi\rangle \otimes \beta_{00} = (\alpha|0\rangle + \beta|1\rangle)\left(\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)\right)$$

$$= \frac{1}{\sqrt{2}}(\alpha|0\rangle(|00\rangle + |11\rangle) + \beta|1\rangle(|00\rangle + |11\rangle))$$

$$\xrightarrow{CNOT(1,2)} \frac{1}{\sqrt{2}}(\alpha|0\rangle(|00\rangle + |11\rangle) + \beta|1\rangle(|10\rangle + |01\rangle))$$

$$\xrightarrow{H(1)} \frac{1}{\sqrt{2}}\left(\alpha\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)(|00\rangle + |11\rangle) + \beta\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)(|10\rangle + |01\rangle)\right)$$

$$\begin{aligned} = \frac{1}{2}[&|00\rangle(\alpha|0\rangle + \beta|1\rangle) \\ &+ |01\rangle(\alpha|1\rangle + \beta|0\rangle) \\ &+ |10\rangle(\alpha|0\rangle - \beta|1\rangle) \\ &+ |11\rangle(\alpha|1\rangle - \beta|0\rangle)] \end{aligned}$$

$$= \frac{1}{2}\sum_{b_1 b_2=0}^{1} |b_1 b_2\rangle \underbrace{(X^{b_2}Z^{b_1})|\psi\rangle}_{3^{er}\ qubit}.$$

Por lo tanto, aplicando $Z^{b_1}X^{b_2}$, Bob obtendrá el estado original $|\psi\rangle$.

Nota 2.42 *Para escribir la compuerta* —$\boxed{Z^{b_1}X^{b_2}}$— *como dos compuertas, se debe hacer como* —$\boxed{X^{b_2}}$—$\boxed{Z^{b_1}}$—*, ya que primero se aplicará la matriz* X^{b_2} *y luego* Z^{b_1}*.*

2.8. Paralelismo Cuántico

2.8.1. Introducción

Consideremos una función $f : \{0,1\} \to \{0,1\}$. Si en una computadora clásica se quiere evaluar esta función. se debe hacer el cálculo para todas las entradas posibles ($f(0)$ y $f(1)$, en este caso).

Consideremos ahora una compuerta cuántica U_f de $\mathbb{C}^4$, que actúe de la siguiente manera

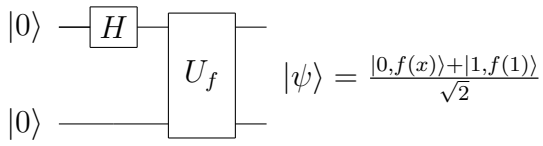
$$U_f \left|x, y\right\rangle = \left|x, y \oplus f(x)\right\rangle,$$

donde $\oplus$ simboliza la suma módulo 2.

Por la definición anterior se tiene que

$$U_f \left|x, 0\right\rangle = \left|x, f(x)\right\rangle.$$

Ahora, consideremos el siguiente circuito:



Veamos

$$|00\rangle \xrightarrow{H(1)} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$

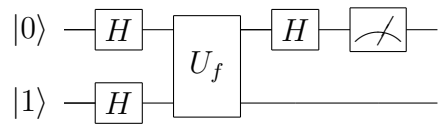
$$\xrightarrow{U_f} \frac{1}{\sqrt{2}}(|0, f(0)\rangle + |1, f(1)\rangle).$$

La salida de este circuito es estado en superposición de todos los resultados posibles de la aplicación de la función f. En principio esta no sería una idea muy práctica, ya que no se puede saber un valor particular de f.

2.8.2. Algoritmo de Deutsch

El objetivo de este algoritmo es saber si una función es constante o no.

Se representa el algoritmo con el siguiente circuito:



$$|01\rangle \xrightarrow{H(1,2)} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |x\rangle\,\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle),$$

donde $|x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.

Veamos qué sucede con la aplicación de la compuerta U_f para cada una de las posibilidades:

$$U_f\,|x,0\rangle = \frac{1}{\sqrt{2}}(|0,f(0)\rangle + |1,f(1)\rangle),$$
$$U_f\,|x,1\rangle = \frac{1}{\sqrt{2}}(|0,1\oplus f(0)\rangle + |1,1\oplus f(1)\rangle),$$

por lo tanto

$$U_f\,|x\rangle\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = \frac{1}{\sqrt{2}}U_f(|x,0\rangle - |x,1\rangle) = \frac{1}{\sqrt{2}}\left(U_f\,|x,0\rangle - U_f\,|x,1\rangle\right)$$

$$= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}(|0,f(0)\rangle + |1,f(1)\rangle) - \frac{1}{\sqrt{2}}(|0,1\oplus f(0)\rangle + |1,1\oplus f(1)\rangle)\right)$$

$$= \frac{1}{2}\left(|0,f(0)\rangle + |1,f(1)\rangle - |0,1\oplus f(0)\rangle - |1,1\oplus f(1)\rangle\right). \tag{2.1}$$

Entonces, si $f(0)\neq f(1)$

$$(2{,}1) = \pm\frac{1}{2}\left(|00\rangle + |11\rangle - |01\rangle - |10\rangle\right) = \pm\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right),$$

y si $f(0)=f(1)$

$$(2{,}1) = \pm\frac{1}{2}\left(|00\rangle + |10\rangle - |01\rangle - |11\rangle\right) = \pm\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right).$$

Luego, se aplica la última compuerta Hadamard y se obtiene lo siguiente

$$\begin{cases} \text{Si } f(0)=f(1) \xrightarrow{H(1)} \pm|0\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \\ \text{Si } f(0)\neq f(1) \xrightarrow{H(1)} \pm|1\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right] \end{cases}.$$

Por lo tanto, uniendo las salidas posibles, se obtiene

$$\pm\,|f(0)\oplus f(1)\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right].$$

Entonces, midiendo el primer qubit se puede saber si los valores de f son iguales o distintos entre sí.

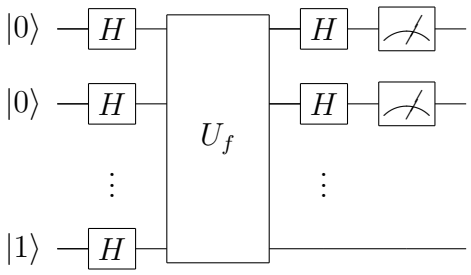
Hasta aquí no se obtiene demasiada *ganancia* con respecto a un algoritmo clásico, ya que clásicamente bastaba con 3 operaciones (evaluar la función en las 2 entradas posibles y compararlas).

En la siguiente subsección se presenta una modificación a este algoritmo que muestra la verdadera *ganancia* con respecto a su contrapartida clásica.

2.8.3. Algoritmo de Deutsch-Jotza

Este algoritmo es una generalización del anterior. Sea f una función que toma n bits y devuelve 0 ó 1. Se quiere saber si es constante o balanceada (*i.e.* que devuelve 0 para la mitad de las entradas posibles y 1 para la otra mitad).

Consideremos el siguiente circuito



La entrada de este algoritmo es $|0\rangle^{\otimes n}|1\rangle = |0\ldots01\rangle$. Aplicando las $n+1$ compuertas Hadamard sobre la entrada, se obtiene

$$\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)^{\otimes n}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)=\sum_{\overline{x}\in\{0,1\}^n}\frac{|\overline{x}\rangle}{\sqrt{2^n}}\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right].$$

Aquí la compuerta U_f es una generalización del caso anterior y se comporta de la siguiente manera

$$U_f|\overline{x},y\rangle=|\overline{x},y\oplus f(\overline{x})\rangle\,.$$

Entonces

$$U_f|\overline{x},0\rangle=|\overline{x},f(\overline{x})\rangle\,,$$

$$\text{y } U_f|\overline{x},1\rangle=|\overline{x},1\oplus f(\overline{x})\rangle\,.$$

Por lo tanto

$$U_f\left(\sum_{\overline{x}\in\{0,1\}^n}\frac{|\overline{x}\rangle}{\sqrt{2^n}}\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]\right)=$$

$$\sum_{\overline{x}\in\{0,1\}^n}\frac{1}{\sqrt{2^n}}U_f|\overline{x}\rangle\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$$

$$= \sum_{\overline{x}\in\{0,1\}^n} \frac{1}{\sqrt{2^{n+1}}} \left(U_f \left|\overline{x}, 0\right\rangle - U_f \left|\overline{x}, 1\right\rangle\right) =$$

$$\sum_{\overline{x}\in\{0,1\}^n} \frac{1}{\sqrt{2^{n+1}}} \left(\left|\overline{x}, f(\overline{x})\right\rangle - \left|\overline{x}, 1 \oplus f(\overline{x})\right\rangle\right)$$

$$= \sum_{\overline{x}\in\{0,1\}^n} \frac{1}{\sqrt{2^n}} \left|\overline{x}\right\rangle \left(\frac{\left|f(\overline{x})\right\rangle - \left|1 \oplus f(\overline{x})\right\rangle}{\sqrt{2}}\right).$$

Notemos que

$$\left.\begin{array}{l} H\left|0\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle) \\ H\left|1\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle) \end{array}\right\} \Rightarrow H\left|y\right\rangle = \frac{1}{\sqrt{2}} \sum_{z\in\{0,1\}} (-1)^{yz} \left|z\right\rangle,$$

por lo tanto

$$H^{\otimes n} \left|\overline{x}\right\rangle = H^{\otimes n} \left|x_1 \ldots x_n\right\rangle$$

$$= \left(\frac{1}{\sqrt{2}} \sum_{z_1\in\{0,1\}} (-1)^{x_1 z_1} \left|z_1\right\rangle\right) \cdots \left(\frac{1}{\sqrt{2}} \sum_{z_n\in\{0,1\}} (-1)^{x_n z_n} \left|z_n\right\rangle\right)$$

$$= \frac{1}{\sqrt{2^n}} \sum_{\overline{z}\in\{0,1\}^n} (-1)^{\overline{x}\cdot\overline{z}} \left|\overline{z}\right\rangle,$$

donde $\overline{x} \cdot \overline{z} = x_1 z_1 + \ldots + x_n z_n$.

Ahora sí, se aplican las n compuertas Hadamard restantes, obteniendo lo siguiente:

$$\overset{H(1,\ldots,n)}{\longrightarrow} \sum_{\overline{x}\in\{0,1\}^n} \frac{1}{\sqrt{2^n}} \left(\frac{1}{\sqrt{2^n}} \sum_{\overline{z}\in\{0,1\}^n} (-1)^{\overline{x}\cdot\overline{z}} \left|\overline{z}\right\rangle\right) \left(\frac{\left|f(\overline{x})\right\rangle - \left|1 \oplus f(\overline{x})\right\rangle}{\sqrt{2}}\right)$$

$$= \sum_{\overline{x}\in\{0,1\}^n} \sum_{\overline{z}\in\{0,1\}^n} \frac{(-1)^{\overline{x}\cdot\overline{z}} \left|\overline{z}\right\rangle}{2^n} \left(\frac{\left|f(\overline{x})\right\rangle - \left|1 \oplus f(\overline{x})\right\rangle}{\sqrt{2}}\right). \tag{2.2}$$

Analicemos este resultado.

- Si f es constante

$$(2,2) = \pm \sum_{\overline{x}\in\{0,1\}^n} \sum_{\overline{z}\in\{0,1\}^n} \frac{(-1)^{\overline{x}\cdot\overline{z}} \left|\overline{z}\right\rangle}{2^n} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right),$$

y en los términos en que $\overline{z} = 0$, los primeros n qubits son

$$\pm \sum_{\overline{x}\in\{0,1\}^n} \frac{|0\rangle^{\otimes n}}{2^n} = \pm\frac{2^n}{2^n} |0\rangle^{\otimes n} = \pm |0\rangle^{\otimes n},$$

por lo tanto los términos en que $\overline{z} \neq 0$ se deberán anular por la condición de normalidad, quedando el siguiente resultado:

$$\pm |0\rangle^{\otimes n} \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right],$$

lo que nos dice que si se miden los primeros n qubits se obtiene $|0\rangle^{\otimes n}$.

- Si f no es constante (50 % de las veces devuelve 0 y 50 % devuelve 1), entonces para $\overline{z} = 0$:

$$(2,2) = \sum_{\overline{x}\in\{0,1\}^n} \frac{|0\rangle^{\otimes n}}{2^n} \left(\frac{|f(\overline{x})\rangle - |1 \oplus f(\overline{x})\rangle}{\sqrt{2}}\right) =$$

$$\sum_{\overline{x}\in\{0,1\}^n} (-1)^{\overline{x}} \frac{|0\rangle^{\otimes n}}{2^n} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = 0,$$

por lo tanto al realizar la medición de los primeros n qubits se obtiene algo distinto de $|0\rangle^{\otimes n}$.

Conclusión: Si se obtiene $|0\rangle^{\otimes n}$ a la salida de la medición, la función es constante, en otro caso la función es balanceada.

2.9. Algoritmo de Búsqueda de Grover

2.9.1. Oráculo

Sea la compuerta U_f, tal que

$$U_f \left|x, b\right\rangle = \left|x, b \oplus f(x)\right\rangle .$$

Si tomamos $b = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, entonces

$$U_f \left|x, b\right\rangle = U_f \left[\left|x\right\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right] = \frac{1}{\sqrt{2}} \left[U_f \left|x, 0\right\rangle - U_f \left|x, 1\right\rangle\right]$$

$$= \frac{1}{\sqrt{2}}(|x, f(x)\rangle - |x, 1 \oplus f(x)\rangle) = \left|x\right\rangle \frac{1}{\sqrt{2}}(|x, f(x)\rangle - |x, 1 \oplus f(x)\rangle)$$

$$= (-1)^{f(x)} \left|x, b\right\rangle .$$

Notemos que U_f no modifica el estado b, por lo tanto podemos omitirlo y referirnos a esta transformación como

$$U \left|x\right\rangle = (-1)^{f(x)} \left|x\right\rangle ,$$

a la cual se le llama *Oráculo*.

2.9.2. Inversión sobre el promedio

Sea el estado $|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle$. Definimos la transformación *Inversión sobre el promedio* a $G = 2 \left|\phi\right\rangle \left\langle\phi\right| - I$.

Entonces

$$G = 2 \left|\phi\right\rangle \left\langle\phi\right| - I = 2 \begin{pmatrix} \frac{1}{\sqrt{2^n}} \\ \vdots \\ \frac{1}{\sqrt{2^n}} \end{pmatrix}_{2^n} \left(\frac{1}{\sqrt{2^n}} \cdots \frac{1}{\sqrt{2^n}}\right)_{2^n} - I$$

$$= \begin{pmatrix} \frac{2}{2^n} - 1 & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} - 1 & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} - 1 \end{pmatrix}_{2^n \times 2^n} .$$

Veamos cómo actúa G sobre un estado cualquiera. Consideremos el estado

$$|\psi\rangle = \sum_{x \in \{0,1\}^n} a_x \left|x\right\rangle,$$

entonces

$$\begin{array}{c|c} G\left|\psi\right\rangle & \begin{pmatrix} a_0 \\ \vdots \\ a_{2^n-1} \end{pmatrix} \\ \hline \begin{pmatrix} \frac{2}{2^n}-1 & \cdots & \frac{2}{2^n} \\ \vdots & & \vdots \\ \frac{2}{2^n} & \cdots & \frac{2}{2^n}-1 \end{pmatrix} & \begin{pmatrix} \left(\sum_{x\in\{0,1\}^n} \frac{2a_x}{2^n}\right) - a_0 \\ \vdots \\ \left(\sum_{x\in\{0,1\}^n} \frac{2a_x}{2^n}\right) - a_{2^n-1} \end{pmatrix} \end{array}.$$

Utilizando notación Dirac, lo escribimos de la siguiente manera:

$$\sum_{x\in\{0,1\}^n} a_x \left|x\right\rangle \xrightarrow{G} \sum_{x\in\{0,1\}^n} \left[\left(\sum_{y\in\{0,1\}^n} \frac{2a_y}{2^n}\right) - a_x\right] \left|x\right\rangle$$

$$= \sum_{x\in\{0,1\}^n} (2A - a_x) \left|x\right\rangle,$$

donde A es el promedio de los a_x.

2.9.3. El algoritmo

Partimos de una lista de tamaño N. Supondremos, incrementando la lista si es necesario, que $N = 2^n$ para algún n. Trabajaremos con los índices de los elementos de la lista, es decir con $x = 0 \ldots 2^n - 1$ y queremos localizar el x_0 tal que $f(x_0) = 1$ para cierta función booleana f.

El input de este algoritmo es $\left|0\right\rangle^{\otimes n}$.

Paso 1: Aplicamos $H^{\otimes n}$:

$$\left|0\right\rangle^{\otimes n} \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} \left|x\right\rangle = \left|\psi_1\right\rangle.$$

Aquí tenemos representados todos los registros de la lista. La idea es subir la probabilidad de que al medir este estado, obtengamos el elemento x_0.

Paso 2: Aplicamos el oráculo:

$$\left|\psi_1\right\rangle \xrightarrow{U} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left|x\right\rangle = \left|\psi_2\right\rangle.$$

Paso 3: Hacemos una inversión sobre el promedio:

$$|\psi_2\rangle = \sum_{x\in\{0,1\}^n} \underbrace{\left[\frac{(-1)^{f(x)}}{\sqrt{2^n}}\right]}_{a_x} |x\rangle \xrightarrow{G} \sum_{x\in\{0,1\}^n} (2A - a_x)\,|x\rangle$$

$$= \sum_{x\in\{0,1\}^n} \left[\left(2\sum_{y\in\{0,1\}^n} \frac{(-1)^{f(y)}}{2^n\sqrt{2^n}}\right) - \frac{(-1)^{f(x)}}{\sqrt{2^n}}\right] |x\rangle$$

$$= \sum_{x\in\{0,1\}^n} \left[\left(2\sum_{\substack{y\in\{0,1\}^n \\ y\neq x}} \frac{(-1)^{f(y)}}{2^n\sqrt{2^n}}\right) + \frac{2(-1)^{f(x)}}{2^n\sqrt{2^n}} - \frac{(-1)^{f(x)}}{\sqrt{2^n}}\right] |x\rangle$$

$$= \sum_{x\in\{0,1\}^n} \left[\left(2\sum_{\substack{y\in\{0,1\}^n \\ y\neq x}} \frac{(-1)^{f(y)}}{2^n\sqrt{2^n}}\right) + \frac{2-2^n}{2^n\sqrt{2^n}}(-1)^{f(x)}\right] |x\rangle.$$

Analicemos este resultado. El término donde $x = x_0$ $(f(x) = 1)$ queda de la siguiente manera

$$\left[\left(2\sum_{\substack{y\in\{0,1\}^n \\ y\neq x_0}} \frac{1}{2^n\sqrt{2^n}}\right) + \frac{2^n-2}{2^n\sqrt{2^n}}\right] |x_0\rangle = \left[\frac{2}{2^n\sqrt{2^n}}(2^n-1) + \frac{2^n-2}{2^n\sqrt{2^n}}\right] |x_0\rangle$$

$$= \left[\frac{2^{n+1}+2^n-4}{2^n\sqrt{2^n}}\right] |x_0\rangle\,,$$

y los términos donde $x \neq x_0$ quedan

$$\left[\left(2\sum_{\substack{y\in\{0,1\}^n \\ y\neq x_0, y\neq x}} \frac{1}{2^n\sqrt{2^n}}\right) + \frac{2(-1)}{2^n\sqrt{2^n}} + \frac{2-2^n}{2^n\sqrt{2^n}}\right] |x_0\rangle$$

$$= \left[\frac{2^{n+1}-2^n-4}{2^n\sqrt{2^n}}\right] |x_0\rangle\,.$$

Como se puede apreciar, el proceso ha cambiado las amplitudes del estado, pasando de tener todas las amplitudes iguales, a tener un estado donde el resultado que nos interesa tiene mayor amplitud que el resto.

Repitiendo este proceso (pasos 2 y 3) se va *levantando* la amplitud del estado que queremos obtener tras la medición. Pasado cierto número de repeticiones, esa amplitud vuelve a decrecer (en la siguiente sección veremos cómo calcular el número óptimo de repeticiones). Cuando la amplitud de x_0 es máxima realizamos una medición, obteniendo con la máxima probabilidad posible el estado x_0.

Ejemplo 2.43 *Supongamos que tenemos una lista de* 16 *elementos, de los que sólo uno, al cual denominaremos* x_0*, verifica la propiedad* $f(x_0) = 1$*.*

Construimos el estado $|0\rangle^{\otimes 4}$ *y aplicamos* $H^{\otimes 4}$ *obteniendo:*

$$\frac{1}{4} \sum_{x \in \{0,1\}^4} |x\rangle.$$

Inicialmente todas las amplitudes son iguales a $\frac{1}{4}$*. Aplicamos el oráculo y obtenemos:*

$$\frac{1}{4} \sum_{x \in \{0,1\}^4} (-1)^{f(x)} |x\rangle.$$

Luego, hacemos la inversión de promedio y la nueva amplitud de x_0 *será:*

$$\frac{2^5 + 2^4 - 4}{2^4\sqrt{2^4}} = \frac{11}{16} = 0{,}6875,$$

mientras que para el resto de los x *la amplitud será*

$$\frac{2^5 - 2^4 - 4}{2^4\sqrt{2^4}} = \frac{3}{16} = 0{,}1875$$

La siguiente tabla muestra las amplitudes y probabilidades de error para las sucesivas iteraciones.

Repetición	*Amplitud de* x_0	*Amplitud de* $x \neq x_0$	*Prob. de error*
1	*0.6875*	*0.1875*	*0.527*
2	*0.953125*	*0.078125*	*0.092*
3	*0.98046875*	*-0.05078125*	*0.039*

Si seguimos iterando empieza a subir la probabilidad de error. Por lo tanto el número óptimo de iteraciones es 3 *y se consigue una probabilidad de error de* 0,039*.*

2.9.4. Cálculo del número óptimo de iteraciones

Luego de k iteraciones, x_0 tendrá una amplitud b_k y el resto tendrán todos una amplitud m_k. Podemos representar este estado como sigue:

$$b_k \left|x_0\right\rangle + m_k \sum_{\substack{x\in\{0,1\}^n \\ x\neq x_0}} \left|x\right\rangle .$$

En cada iteración se aplica U, el cual cambia el signo de b_k, y luego G, por lo tanto se cumplen las siguientes ecuaciones recursivas:

$$\begin{cases} m_0 = b_0 = \frac{1}{\sqrt{2^n}} \\ m_{k+1} = 2A_k - m_k \\ b_{k+1} = 2A_k + b_k \end{cases},$$

donde

$$A_k = \frac{(2^n - 1)m_k - b_k}{2^n}.$$

Se puede demostrar por inducción que las fórmulas cerradas para estas recursiones son

$$m_k = \frac{1}{\sqrt{2^n - 1}} cos((2k+1)\gamma),$$

$$b_k = sen((2k+1)\gamma),$$

donde

$$cos(\gamma) = \sqrt{\frac{2^n - 1}{2^n}}$$

$$\text{y } sen(\gamma) = \sqrt{\frac{1}{2^n}}$$

Para conseguir la mínima probabilidad de error, se debe minimizar $|m_k|$.
Y sabemos que

$$m_k = 0 \Leftrightarrow (2k+1)\gamma = \frac{\pi}{2} \Leftrightarrow k = \frac{\pi}{4\gamma} - \frac{1}{2}.$$

Dado que k debe ser entero, se toma

$$\tilde{k} = \left\lfloor \frac{\pi}{4\gamma} \right\rfloor .$$

Notemos que $|k - \tilde{k}| \leq \frac{1}{2}$, entonces

$$|\frac{\pi}{2} - (2\tilde{k}+1)\gamma| = |(2k+1)\gamma - (2\tilde{k}+1)\gamma| = |2\gamma(k-\tilde{k})| \leq \gamma.$$

Esto sirve para calcular una cota de la probabilidad de error.

La probabilidad de error luego de $\tilde{k}$ iteraciones es

$$(2^n - 1)(m_k)^2 = cos^2((2\tilde{k}+1)\gamma) = sen^2(\frac{\pi}{2} - (2\tilde{k}+1)\gamma) \leq sen^2(\gamma) = \frac{1}{2^n}.$$

$\therefore$ La probabilidad obtener un resultado erroneo luego de $\tilde{k}$ iteraciones es menor a $\frac{1}{2^n}$.

En el ejemplo anterior

$$\tilde{k} = \left\lfloor \frac{\pi}{4asen(\sqrt{\frac{1}{16}})} \right\rfloor = 3,$$

y la probabilidad de error es $0{,}039 \leq \frac{1}{2^4} = 0{,}0625$.

2.10. Resumen del Capítulo

En este capítulo se han presentado las nociones básicas de la Computación Cuántica: la manera tradicional de representar los algoritmos mediante circuitos, los conceptos cruciales como el enredo cuántico, el no-cloning, el paralelismo y la medición. Además se han dado varios ejemplos que ayudan a comprender dichos conceptos en situaciones prácticas.

Para una introducción más extensa al mundo de la Computación Cuántica se aconseja el excelente libro de Michael Nielsen e Isaac Chuang[21].

Los capítulos siguientes utilizarán los conceptos aquí presentados para introducir nuevas formas de representación de los algoritmos.

Capítulo 3

El λ-Cálculo de van Tonder

3.1. Introducción

El área de los lenguajes de programación cuánticos intenta proveer un mecanismo de alto nivel para razonar con los conceptos clave de la computación cuántica, como ser el paralelismo, la medición y el enredo (ver Capítulo 2).

Por otro lado, las técnicas provenientes de la lógica lineal, la teoría de las probabilidades y la aplicación de semántica otorga una nueva perspectiva para razonar con la mecánica cuántica.

La investigación en esta área es reciente pero extensa. El lector interesado en conocer los desarrollos existentes puede consultar las investigaciones realizados por Peter Selinger[23] y Simon Gay[15] en los cuales dan una vista general de los trabajos producidos en este sentido.

En particular nos detendremos en el trabajo de André van Tonder[28], que es la razón de esta tesis. van Tonder define un λ-cálculo lineal con historial haciendo posible de este modo representar los conceptos mencionados anteriormente. En dicho formalismo no se incluye la operación medición (ver definición 2.30), por lo cual los algoritmos deben reescribirse para diferirla hacia el final. Si bien van Tonder afirma que esto es posible hacerlo en todos los casos, el modelo no está completo, y lo que es aún peor: se pierde la comprensión de los algoritmos, que era la primera razón para diseñar un λ-cálculo cuántico.

Volveremos con este análisis en la Sección 3.8, luego de presentar el modelo y algunos ejemplos clave.

3.2. Reversibilidad

Como se mostró en el Capítulo anterior, todas las compuertas cuánticas son reversibles. En el λ-cálculo *clásico*, las β-reducciones consumen las expresiones para entregar un resultado, y en cada paso se descarta información, haciendo el proceso irreversible.

Charles Bennett demostró[8] que cualquier proceso de cómputo puede transformarse en

uno reversible. Por lo tanto, se debe encontrar algún método adecuado para lograr la reversibilidad en las β-reducciones.

Una forma simple de lograr ésto podría ser llevando en un *historial* qué operación se ha realizado y qué subexpresión se ha reducido.

Ejemplo 3.1 *Sea x un término y $\beta : x \to \beta(x)$ la β-reducción de un subtérmino de x. Entonces se puede considerar la función $x \to (x, \beta(x))$, la cual claramente es invertible ya que no descarta el término original.*

El proceso de reducción se podría pensar de la siguiente manera:

$$x \to (x, \beta(x)) \to (x, \beta(x), \beta^2(x)) \to (x, \beta(x), \beta^2(x), \beta^3(x)) \to \dots$$

Veremos que el ejemplo anterior no es suficiente para el caso cuántico sin algunas modificaciones. Para entender el problema, introducimos un nuevo ejemplo:

Ejemplo 3.2 *Consideremos una sintaxis para el λ-cálculo cuántico a la que se le han agregado constantes que denotan qubits y compuertas como se muestra en la Figura 3.1.*

$t ::=$	término
$\quad x$	variable
$\quad (\lambda x.t)$	abstracción
$\quad (t\ t)$	aplicación
$\quad c$	constante
$c ::=$	
$\quad 0 \mid 1 \mid H \mid cnot \mid X \mid Z \mid \dots$	constantes

Figura 3.1: Sintaxis para el ejemplo 3.2. 0 y 1 denotan primitivas, el resto de las constantes denotan compuertas básicas entre qubits.

Sea un estado inicial representado por el string $|(H\ 0)\rangle$. La idea es elegir reglas de transición adecuadas para que este estado evalúe a una compuerta Hadamard aplicada a $|0\rangle$. Una regla candidata podría ser

$$|(H\ 0)\rangle \to \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle),$$
$$|(H\ 1)\rangle \to \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Y se puede usar el truco *para hacerlo reversible:*

$$|(H\ 0)\rangle \to \frac{1}{\sqrt{2}}(|(H\ 0); 0\rangle + |(H\ 0); 1\rangle)$$
$$= |(H\ 0)\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

El ";" denota la concatenación de strings y de alguna manera determinamos que se puede factorizar *como se ha hecho en la última expresión.*

En este caso se tiene un historial ($|(H\ 0)\rangle$) y un resultado de la evaluación de dicho término ($\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$).

El problema surge con ejemplos un poco más complejos, donde el historial y el registro computacional quedan enredados.

Considérese, por ejemplo, la evaluación del string $|(H\ (H\ 0))\rangle$:

$$|(H(H\ 0))\rangle \to \frac{1}{\sqrt{2}}(|H(H\ 0); (H\ 0)\rangle + |H(H\ 0); (H\ 1)\rangle$$

$$\to \frac{1}{2}\,|(H(H\ 0))\rangle \otimes (|(H\ 0); 0\rangle + |(H\ 0); 1\rangle + |(H\ 1); 0\rangle - |(H\ 1); 1\rangle).$$

En este último término, el historial y el registro computacional han quedado enredados.

El problema mostrado en el ejemplo anterior surge porque se está guardando en el historial más información de la realmente necesaria. Para lograr reversibilidad, sólo se necesita guardar qué subtérmino se ha reducido y qué operación se ha aplicado.

Ejemplo 3.3

$$|(H(H\ 0))\rangle \to \frac{1}{\sqrt{2}}(|\_(H\ \_); (H\ 0)\rangle + |\_(H\ \_); (H\ 1)\rangle)$$

$$\to \frac{1}{2}\,|(\_(H\ \_))\rangle \otimes (|(H\ \_); 0\rangle + |(H\ \_); 1\rangle + |(H\ \_); 0\rangle - |(H\ \_); 1\rangle)$$
$$= \frac{1}{2}\,|(\_(H\ \_))\rangle \otimes 2\,|(H\ \_); 0\rangle = |(\_(H\ \_)); (H\ \_)\rangle \otimes |0\rangle$$

En cada paso se reemplazan los subtérminos que no son necesarios para la reversibilidad por el placeholder "_".

Para formalizar las ideas expuestas, primero se extiende la definición de valores para incluir a las constantes (ver Figura 3.2), y el estado computacional se toma como una superposición cuántica de secuencias de la forma

$$h_1; \ldots; h_n; t,$$

$v ::=$	valores
x	variable
c	constante
$(\lambda x.t)$	valor de abstracción

Figura 3.2: Definición de valores para el λ-cálculo con historial.

donde a $h_1; \ldots; h_n$ se le llama *historial* y a t *registro computacional*.

Las reglas de transición se detallan en la Figura 3.3. En dichas reglas se ha agregado el manejo del historial para que cada paso agregue la información necesaria para hacer reversible el cómputo.

$\bar{t}_x$ se obtiene de t reemplazando recursivamente todos los subtérminos que no contienen x con el símbolo $\_$ y manteniendo x (ver definición en Figura 3.4).

$$\frac{t_1 \to h_1; t_1'}{\mathcal{H}; (t_1\ t_2) \to \mathcal{H}; (h_1\ \_); (t_1'\ t_2)} \quad (APP_1)$$

$$\frac{t_2 \to h_2; t_2'}{\mathcal{H}; (v_1\ t_2) \to \mathcal{H}; (\_\ h_2); (v_1\ t_2')} \quad (APP_2)$$

$$\frac{}{\mathcal{H}; ((\lambda x.t)\ v) \to \mathcal{H}; ((\lambda x.\bar{t}_x); \_); t[v/x]} \quad (\beta_1)\ Si\ x \in F(t)$$

$$\frac{}{\mathcal{H}; ((\lambda x.t)\ v) \to \mathcal{H}; ((\lambda x.\_); v); t} \quad (\beta_2)\ Si\ x \notin F(t)$$

$$\frac{}{|\mathcal{H}; (c_U\ \phi)\rangle \to |\mathcal{H}; (c_U\ \_)\rangle \otimes U\,|\phi\rangle} \quad (U)$$

$$\frac{}{\mathcal{H}; t \to \mathcal{H}; \_; t} \quad (Id)\ en\ otro\ caso$$

Figura 3.3: Reglas de transición para el λ-cálculo con historial.

Ejemplo 3.4 $|((apply\ id)\ cosa)\rangle \equiv |(((\lambda f.(\lambda x.(f\ x)))\ (\lambda z.z))\ cosa)\rangle$
$\to |(((\lambda f.(\_.(f\ \_)))\_)\_); (\lambda x.((\lambda z.z)\ x)\ cosa)\rangle$
$\to |(((\lambda f.(\_.(f\ \_)))\_)\_); (\lambda x.(\_\ x)\_); ((\lambda z.z)\ cosa)\rangle$
$\to |(((\lambda f.(\_.(f\ \_)))\_)\_); (\lambda x.(\_\ x)\_); ((\lambda z.z)\ \_);\ cosa\rangle$
$\to |(((\lambda f.(\_.(f\ \_)))\_)\_); (\lambda x.(\_\ x)\_); ((\lambda z.z)\ \_); \_;\ cosa\rangle$
$\to \ldots$

$$\begin{array}{rcl} \overline{t}_x & \equiv & \_ \text{ si } x \notin F(t) \\ \overline{(\lambda y.t)}_x & \equiv & (\_\ \overline{t}_x) \\ \overline{(t\ t')}_x & \equiv & (\overline{t}_x\ \overline{t'}_x) \\ \overline{x}_x & \equiv & x \end{array}$$

Figura 3.4: Definición de $\overline{t}$.

En este ejemplo se puede usar como criterio de terminación comparar la última expresión con "_", pero esto no es del todo correcto ya que el estado podría tener una superposición de varios historiales, algunos de los cuales hayan terminado y otros no.

Este y otros problemas se resuelven en la Sección siguiente.

La regla extra, U, es la que indica cómo se deben reducir las expresiones que involucran la aplicación de compuertas cuánticas.

- c_U denota cualquiera de los símbolos para compuertas cuánticas y U la correspondiente transformación unitaria.
- ϕ es 0 ó 1 en el caso de operadores de 1 qubit, $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$ en el caso de operadores de 2-qubits, etc.

Ejemplo 3.5

$$|(cnot(1,0))\rangle \rightarrow |(cnot\ \_); (1,1)\rangle$$
$$|\mathcal{H}; (H\ 0)\rangle \rightarrow |\mathcal{H}; (H\ \_)\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

3.3. Superposición y Enredo

En la Sección precedente se presentó una sintaxis y un modelo operacional que no es del todo correcto. El siguiente ejemplo pondrá de manifiesto el problema.

Ejemplo 3.6 $|((\lambda x.cosa)\ otraCosa)\rangle$
$\rightarrow |((\lambda x.\_)\ otraCosa); cosa\rangle$
Aquí se debe guardar "otraCosa" en el historial para mantener reversibilidad.

El problema surge cuando el argumento que se descarta es una superposición. Los siguientes ejemplos aclararán el panorama.

Ejemplo 3.7 $|(\lambda x.0)\ (H\ 0)\rangle$
$\rightarrow |(\_\ (H\ \_))\rangle \otimes \left|(\lambda x.0)\ (\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))\right\rangle$

Aquí hay dos formas de reducir el registro computacional:

1. $\Big|(\lambda x.\_)\ (\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\Big\rangle \otimes |0\rangle$
   $\equiv \frac{1}{\sqrt{2}}(|(\lambda x.\_)\ 0\rangle + |(\lambda x.\_)\ 1\rangle) \otimes |0\rangle$

2. $\frac{1}{\sqrt{2}}(|(\lambda x.0)\ 0\rangle + |(\lambda x.0)\ 1\rangle)$
   $\to \sqrt{2}\,|0\rangle$

   Este estado no representa un qubit, ya que no está normalizado.

Un ejemplo con un grado más de complejidad es el siguiente:

Ejemplo 3.8 $|((\lambda y.((\lambda x.y)\ y))\ (H\ 0))\rangle$
$\to \frac{1}{\sqrt{2}}\,|(\_(H\ \_)\rangle \otimes (|((\lambda y.((\lambda x.y)\ y))\ 0)\rangle + |((\lambda y.((\lambda x.y)\ y))\ 1)\rangle)$
$\to \frac{1}{\sqrt{2}}\,|(\_(H\ \_)\rangle \otimes |((\lambda y.((\_.y)\ y))\ \_)\rangle \otimes (|((\lambda x.0)\ 0)\rangle + |((\lambda x.1)\ 1)\rangle)$
$\to \frac{1}{\sqrt{2}}\,|(\_(H\ \_)\rangle \otimes |((\lambda y.((\_.y)\ y))\ \_)\rangle \otimes (|((\lambda x.\_)\ 0); 0\rangle + |((\lambda x.\_)\ 1); 1\rangle)$

Aquí vemos que quedó el historial enredado con el estado.

En esta sección veremos cómo se resuelve este problema.

Definición 3.9 *Se dice que una subexpresión es* definida con respecto a la base computacional *si es textualmente la misma en todos los branches de una superposición.*

Ejemplo 3.10

$$\frac{1}{\sqrt{2}}(|(\lambda x.0)\ 0\rangle + |(\lambda x.0)\ 1\rangle)$$

La subexpresión $(\lambda x.0)$ *es definida, aunque el argumento* $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *no lo es.*

Se considerará un λ-cálculo que guarde información de cuando un argumento es definido o no, el cual hará imposible escribir funciones que descarten elementos no definidos. Existe un tipo de cálculo sensible a la clase de elementos llamado λ-cálculo lineal, que ha sido estudiado extensamente [1][30][19][22]. La sintaxis que se usará será una extensión de la introducida en [30] y puede verse en la Figura 3.5.

Nota 3.11 *Algunas observaciones*

- *Los términos de la forma* $!t$ *son llamados* no lineales. *Los términos no lineales son términos definidos con respecto a la base computacional.*
- *La abstracción* $(\lambda !x.t)$ *denota funciones con argumentos no lineales. En contraposición, en la abstracción* $(\lambda x.t)$ *el argumento es lineal.*

$t ::=$	término
x	variable
$(\lambda x.t)$	abstracción
$(t\ t)$	aplicación
c	constante
$!t$	término *no lineal*
$(\lambda !x.t)$	abstracción *no lineal*
$c ::=$	
$0\|1\|H\|cnot\|X\|Z\|\ldots$	constantes

Figura 3.5: Sintaxis del λ_q.

- *En una abstracción lineal se pueden usar todos los términos no lineales que se quiera (o ninguno), pero* debe *haber un término lineal, y sólo uno, en el cuerpo de la función.*

Para implementar estas reglas, se necesita la noción de términos "*bien-formados*" (ver Figura 3.6). Esto corresponde a la restricción de que los argumentos lineales aparezcan linealmente en el cuerpo de una función y que todas las variables libres de un término $!t$ refieran a variables no lineales.

Ejemplo 3.12 *Términos Bien-Formados*
$(\lambda !x.0)$
$(\lambda x.x)$
$(\lambda !x.(x\ x))$
$(\lambda y.(\lambda !x.y))$
$(\lambda !y.!(\lambda !x.y))$

Ejemplo 3.13 *Términos No bien-formados*
$(\lambda x.0)$
$(\lambda x.!x)$
$(\lambda x.(x\ x))$
$(\lambda y.(\lambda x.y))$
$(\lambda y.!(\lambda !x.y))$

Las restricciones de términos bien-formados impiden escribir funciones que descarten argumentos lineales, pero esto no es suficiente para prevenir cómputos inseguros sin especificar el orden de reducción.

$\dfrac{}{\vdash c}$	*Const*
$\dfrac{}{x \vdash x}$	*Id*
$\dfrac{!x_1, \ldots, !x_n \vdash t}{!x_1, \ldots, !x_n \vdash !t}$	*Pomotion*
$\dfrac{\Gamma, x \vdash t}{\Gamma, !x \vdash t}$	*Dereliction*
$\dfrac{\Gamma, !x, !y \vdash t}{\Gamma, !z \vdash t\,[z/x, z/y]}$	*Contraction*
$\dfrac{\Gamma \vdash t}{\Gamma, !x \vdash t}$	*Weakening*
$\dfrac{\Gamma, x \vdash t}{\Gamma \vdash (\lambda x.\, t)}$	$\multimap$-*I*
$\dfrac{\Gamma, !x \vdash t}{\Gamma \vdash (\lambda !x.\, t)}$	$\to$-*I*
$\dfrac{\Gamma \vdash t_1 \quad \Delta \vdash t_2}{\Gamma, \Delta \vdash (t_1\ t_2)}$	$\multimap$-*E*

Figura 3.6: Reglas para términos bien-formados en λ_q.

Ejemplo 3.14 *La expresión* $((\lambda !x.0)\ !(H\ 0))$ *es bien-formada. El problema está en que se permite usar* ! *para promover la expresión* $(H\ 0)$*, que va a ser descartada, a un valor no lineal. Si se reduce primero el subtérmino* $(H\ 0)$*, razonando ecuacionalmente se tiene*

$$|((\lambda !x.0)\ !(H\ 0)\rangle = \frac{1}{\sqrt{2}}(|((\lambda !x.0)\ !0)\rangle + |((\lambda !x.0)\ !1)\rangle) = \sqrt{2}\,|0\rangle\,.$$

En cambio, si se considera a $!(H\ 0)$ *como irreducible, se podría* β*-reducir inmediatamente y obtener*

$$|((\lambda !x.0)\ !(H\ 0))\rangle = |0\rangle\,.$$

Para prevenir que términos de la forma $!t$ sean evaluados, se sigue a Abramsky[1] y se extiende la definición de valores como muestra la Figura 3.7

$v ::=$	valores:
x	variable
c	constante
$(\lambda x.t)$	abstracción lineal
$(\lambda !x.t)$	abstracción no lineal
$!t$	!-suspensión

Figura 3.7: Valores en el λ_q.

3.4. Modelo Operacional

El modelo operacional se describe en la Figura 3.8 (donde $\bar{t}$ es definido como en la Figura 3.4).

$$\frac{t_1 \to h_1; t_1'}{\mathcal{H}; (t_1\ t_2) \to \mathcal{H}; (h_1\ \_); (t_1'\ t_2)} \quad (APP_1)$$

$$\frac{t_2 \to h_2; t_2'}{\mathcal{H}; (v\ t_2) \to \mathcal{H}; (\_\ h_2); (v_1\ t_2')} \quad (APP_2)$$

$$\frac{}{\mathcal{H}; ((\lambda x.t)\ v) \to \mathcal{H}; ((\lambda x.\bar{t}_x)\ \_); t[v/x]} \quad (\beta)$$

$$\frac{}{\mathcal{H}; ((\lambda !x.t)\ !t') \to \mathcal{H}; ((\lambda !x.\bar{t}_x)\ \_); t[t'/x]} \quad (!\beta_1)\ Si\ x \in F(t)$$

$$\frac{}{\mathcal{H}; ((\lambda !x.t)\ !t') \to \mathcal{H}; ((\lambda !x.\_)\ !t'); t} \quad (!\beta_2)\ Si\ x \notin F(t)$$

$$\frac{}{|\mathcal{H}; (c_U\ \phi)\rangle \to |\mathcal{H}; (c_U\ \_)\rangle \otimes U\,|\phi\rangle} \quad (U)$$

$$\frac{}{\mathcal{H}; t \to \mathcal{H}; \_; t} \quad (Id)\ en\ otro\ caso$$

Figura 3.8: Modelo operacional para el λ_q.

De acuerdo a estas reglas, las superposiciones cuánticas sólo pueden ser creadas mediante la evaluación de términos que contengan primitivas cuánticas. Nótese que cuando una función no lineal encuentra un término lineal, lo único que se puede aplicar es la regla Id.

Estas reglas permiten crear superposición, pero los términos en la superposición sólo difieren en las posiciones que contienen las constantes 0 y 1. A continuación se da una formalización de esto.

Definición 3.15 *Dos términos son* congruentes *si coinciden símbolo a símbolo excepto tal*

vez en las posiciones que contienen 0 *ó* 1.

Lema 3.16 *Todos los términos en una superposición obtenidos mediante una secuencia de reducción de un término inicial definido son congruentes.* □

Lema 3.17 *Si t es bien-formado y $|\mathcal{H}; t\rangle \to \sum_i c_i |\mathcal{H}'_i; t'_i\rangle$, entonces todos los términos t'_i son bien-formados.* □

Dado que los términos que aparecen en una superposición tienen la misma forma, tiene sentido hablar acerca de subtérminos específicos de la expresión en el registro computacional. Por lo tanto, podemos formular el siguiente lema:

Lema 3.18 *Empezando con un término inicial definido, cualquier subtérmino* !*-suspensión que ocurra durante la reducción será definido con respecto a la base computacional.* □

Nota 3.19 *: Mmediante reducción tampoco se puede crear* !*-suspensiones no definidas:*

$$(\lambda x. \cdots !(\cdots x \cdots) \cdots)\ (H\ 0),$$

ya que x es lineal, lo que implica que $!(\cdots x \cdots)$ es un subtérmino no bien-formado.

Lema 3.20 *Dado un término inicial definido, los contenidos del historial se mantienen definidos a través de la reducción.* □

Corolario 3.21 *La terminación se puede testear observando el último término del historial sin modificar nada. Cuando ese término es igual al placeholder "_", el resultado queda en el registro computacional.* □

3.5. Sistema de prueba ecuacional

El hecho de que el historial se mantenga definido en el λ_q elimina los impedimentos para definir una teoría ecuacional. De hecho, como ahora se puede garantizar que cualquier estado computacional será de la forma $|\mathcal{H}\rangle \otimes |c\rangle$ (i.e. no enredado) y que $|\mathcal{H}\rangle$ se mantendrá definido, por lo cual se preservará la normalización, se puede enunciar el siguiente teorema:

Teorema 3.22 *En el cálculo cuántico λ_q, la evolución del registro computacional está gobernada por las reglas de reducción que se listan en la Figura 3.9.* □

Este teorema provee un conjunto de reglas de reducción simple y que nos permite razonar sobre los cómputos sin tener que acarrear el historial.

Para presentar la teoría ecuacional, se lista el conjunto de axiomas y reglas de inferencia de dicha teoría en la Figura 3.10. Como se puede observar, no hay ninguna regla que permita sustituciones dentro de las !-suspensiones.

Teorema 3.23 *En el Lambda Cálculo Cuántico, la evolución del registro computacional procede reemplazando términos por términos iguales de acuerdo a la teoría ecuacional de la Figura 3.10.* □

$$\frac{t_1 \to t_1'}{(t_1\ t_2) \to (t_1'\ t_2)} \quad (APP_1)$$

$$\frac{t_2 \to t_2'}{(v_1\ t_2) \to (v_1\ t_2')} \quad (APP_2)$$

$$\frac{}{(\lambda x.t)\ v \to t[v/x]} \quad (\beta)$$

$$\frac{}{(\lambda !x.t)\ !t' \to t[t'/x]} \quad (!\beta_1)\ Si\ x \in F(t)$$

$$\frac{}{(\lambda !x.t)\ !t' \to t} \quad (!\beta_2)\ Si\ x \notin F(t)$$

$$\frac{}{|c_U\ \phi\rangle \to U\,|\phi\rangle} \quad (U)$$

Figura 3.9: Reglas de reducción del registro computacional.

3.6. Recursión y Punto Fijo

En λ_q se puede definir el operador de Punto Fijo de la siguiente manera:

$$fix \equiv (\underbrace{(\lambda !u.\lambda !f.(f\ !((u\ !u)\ !f)))}_{v}\ !\underbrace{(\lambda !u.\lambda !f.(f\ !((u\ !u)\ !f)))}_{v}) = (v\ !v)$$

Para entender cómo actúa, se puede ver qué sucede al aplicar este operador a un término cualquiera:

$fix\ !t$
$\to (\lambda !f.(f\ !((v\ !v)\ !f)))\ !t$
$\to t\ !((v\ !v)\ !t)$
$\to t\ !(fix\ !t)$

Ejemplo 3.24 *Si* $t \equiv \lambda !f.u$
$fix\ !t$
$\to t\ !(fix\ !t) \equiv (\lambda !f.u)\ !(fix\ !t)$
$\to u[(fix\ !t)/f]$

En otras palabras, $fix\ !t$ *se copia a sí mismo en el cuerpo (*u*) de* t *a través de la reducción.*

$$\frac{}{t = t} \quad \textit{(refl)}$$

$$\frac{t_1 = t_2}{t_2 = t_1} \quad \textit{(sim)}$$

$$\frac{t_1 = t_2 \;\; t_2 = t_3}{t_1 = t_3} \quad \textit{(trans)}$$

$$\frac{t_1 = t_2 \;\; t_3 = t_4}{(t_1\ t_3) = (t_2\ t_4)} \quad \textit{(app)}$$

$$\frac{t_1 = t_2}{\lambda x.t_1 = \lambda x.t_2} \quad (\lambda_1)$$

$$\frac{t_1 = t_2}{\lambda !x.t_1 = \lambda !x.t_2} \quad (\lambda_2)$$

$$\frac{}{(\lambda x.t)\ v = t[v/x]} \quad (\beta)$$

$$\frac{}{(\lambda !x.t)\ !t' = t[t'/x]} \quad (!\beta_1)\ Si\ x \in F(t)$$

$$\frac{}{(\lambda !x.t)\ !t' = t} \quad (!\beta_2)\ Si\ x \notin F(t)$$

$$\frac{}{|c_U\ \phi\rangle = U\,|\phi\rangle} \quad (U)$$

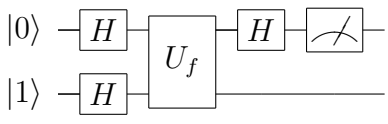
Figura 3.10: Sistema de prueba ecuacional para el cálculo cuántico λ_q

3.7. Ejemplos

3.7.1. Algoritmo de Deutsch

Un algoritmo muy conocido en Computación Cuántica es el llamado *Algoritmo de Deutsch* (ver Subsección 2.8.2).

El circuito de este algoritmo es el siguiente:



donde U_f es una compuerta que actúa de la siguiente manera

$$U_f\,|x, y\rangle = |x, y \oplus f(x)\rangle\,,$$

para alguna función conocida f de 1 bit.

En λ_q se podría escribir así:

$$\begin{array}{rl} \mathsf{deutsch}\ U_f \equiv & \textbf{let}\ \ (x,y) = U_f\ ((H\ 0),(H\ 1))\ \textbf{in} \\ & ((H\ x),y) \end{array}$$

Ejemplo 3.25 *Sea f la función identidad. Entonces, es fácil ver que*

$$\begin{array}{c} U_f\,|00\rangle \rightarrow |00\rangle \\ U_f\,|01\rangle \rightarrow |01\rangle \\ U_f\,|10\rangle \rightarrow |11\rangle \\ U_f\,|11\rangle \rightarrow |10\rangle \end{array}$$

$$\therefore U_f \equiv cnot$$

Entonces

$$\begin{array}{rcl} \mathsf{deutsch}\ cnot & \equiv & \textbf{let}\ \ (x,y) = cnot\ ((H\ 0),(H\ 1))\ \ \textbf{in} \\ & & ((H\ x),y) \end{array}$$

Desarrollando $cnot\ ((H\ 0),(H\ 1))$ se obtiene

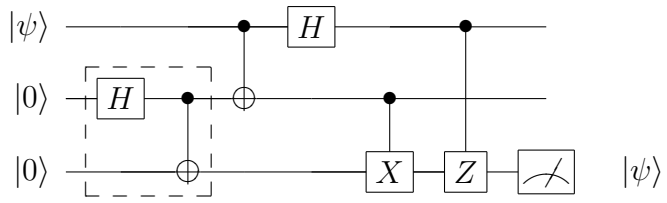
$$\begin{array}{c} cnot\ ((H\ 0),(H\ 1)) \\ = cnot\ (\frac{1}{2}(|0\rangle + |1\rangle), \frac{1}{2}(|0\rangle - |1\rangle)) \\ = cnot\ (\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)) \\ = \frac{1}{2}(|00\rangle - |01\rangle + |11\rangle - |10\rangle) \\ = (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)) \end{array}$$

Reemplazando en el código original se llega a:

$$\begin{array}{c} \textbf{let}\ \ (x,y) = (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle))\ \ \textbf{in} \\ ((H\ x),y) \\ = (H\ (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)) \\ = |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{array}$$

3.7.2. Teleportación

El λ_q no puede representar la operación medición, por lo tanto se debe modificar el algoritmo para diferir la medición. El circuito queda como sigue



Si bien este circuito es correcto, no es del todo claro, pues no es evidente que Alice y Bob quedan separados físicamente durante el proceso. Haremos una discusión más detallada en la Sección 3.8.

El algoritmo se define en λ_q de la siguiente manera:

$$\begin{aligned}\mathsf{teleport}\ x \longrightarrow\ & \textbf{let}\ (e_1, e_2) = \mathsf{epr}\ \textbf{in}\\ & \textbf{let}\ (x', y') = \mathsf{alice}\ (x, e_1)\ \textbf{in}\\ & \ \ \mathsf{bob}\ (x', y', e_2)\end{aligned}$$

donde

$$\mathsf{alice}\ (x, e_1) \longrightarrow\ \textbf{let}\ (x', y') = \mathit{cnot}\ (x, e_1)\ \textbf{in}\ ((H\ x'), y')$$

$$\begin{aligned}\mathsf{bob}\ (x', y', e_2) \longrightarrow\ & \textbf{let}\ (y'', e_2') = cX\ (y', e_2)\ \textbf{in}\\ & \textbf{let}\ (x'', e_2'') = cZ\ (x', e_2')\ \textbf{in}\\ & \ \ (x'', y'', e_2'')\end{aligned}$$

$$\mathsf{epr} \equiv \mathit{cnot}\ ((H\ 0), 0)$$

3.8. Discusión: Importancia de la Medición

Quizá si un físico leyera esta tesis, le llamaría mucho la atención el título de esta sección, ya que cualquier físico encontraría casi absurdas las preguntas ¿Es importante la medición? ¿Qué podríamos hacer sin ella? La medición pasa desapercibida a los ojos de los cientistas de la computación ya que es una operación trivial en la computación clásica, pero aún así esta presente.

Para razonar sobre estos interrogantes, se propone al lector la siguiente situación hipotética: Imagine que alguien afirma haber diseñado una máquina tan eficiente que es capaz de realizar cualquier cálculo, por más complejo que éste sea, en tan sólo un paso, y dicho paso siempre demora sólo unos pocos nanosegundos. *El único detalle* -diría esta persona ficticia- *es que la máquina no cuenta con ninguna herramienta para mostrar el resultado del cálculo.*

¿Podría Ud. afirmar que ésta persona miente? Si no tiene forma de *observar* el resultado, ¿Cómo puede alguien afirmar que no se ha llegado al resultado esperado?

Tal vez el ejemplo anterior sea demasiado irreal, pero da una buena idea de qué tan importante es medir y observar los resultados.

Por supuesto, André van Tonder nunca afirma que se pueda prescindir de la medición, sólo afirma que el hecho de que éste no sea parte del cálculo, no afecta a la comprensión de los algoritmos. Para refutar esta sentencia, volvamos al ejemplo de Teleportación con medición diferida (ver Subsección 3.7.2). En este ejemplo se puede ver que las últimas operaciones (a realizar por Bob) son la aplicación de las compuertas *Controlled-X* y *Controlled-Z*. En ambos casos, el qubit de control lo posee Alice, quien se encuentra distante físicamente de Bob, incluso podría estar a miles de kilómetros de distancia. La única manera de llevar a cabo estas operaciones es que Alice decida enviarle sus qubits a Bob. ¡Pero entonces podría haberle enviado directamente el qubit que se quiere teleportar!

Viéndolo desde otro punto de vista, si revisamos el algoritmo escrito en λ_q, vemos que luego de que Alice realiza sus operaciones, sus qubits son pasados como *argumentos* a Bob, no hay ningún proceso de medición en medio, por lo cual es de esperarse que ésto sólo pueda realizarse enviando *físicamente* los qubits.

Si se contara con la medición, con sólo escribir algo como

$$\begin{array}{l}\textbf{let } (x, y) = M_2 \text{ alice } \textbf{in} \\ \quad \text{bob } (x, y)\end{array}$$

se podría entender a simple vista que Alice mide sus qubits para pasárselos a Bob, y dado que Alice ya ha medido los qubits, puede pasarle a Bob esa información con una simple llamada telefónica, o con cualquier otro canal clásico (en particular, sabemos que el resultado de la medición, al ser qubits de la base, puede encodearse mediante dos bits clásicos).

En el siguiente capítulo se verá de qué manera se puede agregar medición al cálculo y se retomará el ejemplo de Teleportación para validar dicha propuesta.

3.9. Resumen del Capítulo

En este capítulo se ha presentado el modelo de van Tonder y las decisiones tomadas para lograr manejar cada uno de los conceptos de la computación cuántica que no tienen ninguna contrapartida clásica:

- Para obtener *reversibilidad* se ha añadido un historial al cálculo, haciendo que toda transición coloque en el historial el término que se ha reducido y la operación aplicada, de esta manera se puede volver al término original logrando la reversibilidad.
- Para manejar el *enredo cuántico* y evitar que el registro computacional quede enredado con el historial, se han utilizado los conceptos de la lógica lineal y se distingue cuándo un estado es definido y cuándo no.

- Se ha decidido diferir la *medición* y que ésta no sea parte del cálculo. En la sección 3.8 se ha realizado una discusión sobre dicha desición y el siguiente capítulo estará destinado a agregar medición al cálculo

Además, se ha desarrollado una teoría que permite razonar ecuacionalmente la evolución de un cómputo.

Finalmente, se ha presentado un operador de Punto Fijo, completando con ello el cálculo.

Capítulo 4

λ-Cálculo Cuántico con Medición

4.1. Introducción

Como se discutió en el Capítulo precedente (ver Sección 3.8), la operación medición es de suma importancia, no sólo para contar con un cálculo que sea completo, sino también para la comprensión de los algoritmos.

Peter Selinger ha argumentado[23] que para tener medición en un λ-cálculo, sería necesario tiparlo. Esta idea la llevó a la práctica más tarde junto a Benîot Varilon[25].

En este Capítulo mostraremos cómo se puede agregar medición manteniéndose en un λ-cálculo lineal *no tipado*, conservando de esta manera la simplicidad original del modelo de van Tonder.

4.2. Sintaxis

La operación medición tiene en cuenta la *forma* del qubit, por lo tanto, la sintaxis utilizada en el λ_q (Figura 3.5) no es suficiente, ya que ésta engloba en un sólo símbolo de constante a los qubits y a las compuertas, sin especificar su *forma*.

Para poder agregar medición, se deberá distinguir sintácticamente a los qubits de las compuertas y el operador medición. Además, dado que el comportamiento del operador medición es distinto al de las compuertas (recuérdese que la medición no es reversible), éstos se deberán diferenciar también.

El operador medición se escribe sintácticamente como un símbolo constante aislado, con una referencia a la cantidad de qubits a medir

$$M_n.$$

Por otro lado, para formar qubits se utiliza la siguiente sintaxis

$$
\begin{array}{lll}
q ::= & & \\
 & |0\rangle \mid |1\rangle & \text{defined qubits} \\
 & (q \otimes q) & \text{tensor product} \\
 & (q + q) & \text{superposition} \\
 & \alpha(q) & \text{scalar product}
\end{array}
$$

Por último, las compuertas cuánticas se presentan con una constante para ellas

$$c_U ::= H \mid cnot \mid X \mid Z \mid q(q)^T \mid etc.$$

La sintaxis completa se muestra en la Figura 4.1.

$$
\begin{array}{lr}
\ddot{t} ::= & \textit{pre-términos:} \\
\quad x & \textit{variable} \\
\quad (\lambda x.\ddot{t}) & \textit{abstracción} \\
\quad (\ddot{t}\ \ddot{t}) & \textit{aplicación} \\
\quad c_U & \textit{constante para compuertas} \\
\quad q & \textit{qubit} \\
\quad !\ddot{t} & \textit{pre-término no lineal} \\
\quad (\lambda !x.\ddot{t}) & \textit{abstracción no lineal} \\
\quad M_n & \textit{constante para medición} \\
q ::= & \textit{qubits:} \\
\quad |0\rangle \mid |1\rangle & \textit{qubits definidos} \\
\quad (q \otimes q) & \textit{producto tensorial} \\
\quad (q + q) & \textit{superposición} \\
\quad \alpha(q) & \textit{producto escalar} \\
c_U ::= & \textit{constantes para compuertas:} \\
\quad H \mid cnot \mid X \mid Z \mid q(q)^T \mid etc\ \ldots &
\end{array}
$$

Figura 4.1: Sintaxis para pre-términos en el λ_q^M.

Notar que $q(q)^T$, donde $(q)^T$ denota al transpuesto conjugado de q, es una compuerta sólo en algunos casos (i.e. si dicho producto produce una matriz unitaria y hermítica), pero dado

que a las compuertas se les ha dado la libertad de ser constantes, no nos preocuparemos por su forma, de la misma manera que la sintaxis original de van Tonder no tenía en cuenta la forma de las constantes.

Por el contrario, necesitamos determinar cuándo un qubit está bien formado, ya que con la sintaxis aquí descripta, se pueden formar términos como $|0\rangle + |1\rangle$, los cuales no representan ningún estado válido. Por este motivo extenderemos las reglas de bien-formado descriptas en el Capítulo anterior (ver Figura 3.6).

Consideramos a los términos que se generan de la sintaxis de la Figura 4.1 como *pre-términos*, y los términos del lenguaje son los pre-términos que respeten las reglas de bien-formado que se describen en las Figuras 4.2 y 4.3.

$$\frac{}{x \vdash x} \quad Id$$

$$\frac{n \in \mathbb{N}}{\vdash M_n} \quad M$$

$$\frac{}{\vdash c_U} \quad Gate$$

$$\frac{}{\vdash ! |0\rangle} \quad Zero$$

$$\frac{}{\vdash ! |1\rangle} \quad One$$

$$\frac{\Gamma \vdash q_1 \qquad \Delta \vdash q_2}{\Gamma, \Delta \vdash q_1 \otimes q_2} \quad Tensor$$

$$\frac{\Gamma \vdash ! q_1 \qquad \Delta \vdash ! q_2}{\Gamma, \Delta \vdash ! q_1 \otimes ! q_2} \quad !Tensor$$

$$\frac{\sum_{i=1}^{2^n} |\alpha_i|^2 = 1 \qquad \alpha_i \in \mathbb{C}, i = 1, \ldots, 2^n}{\vdash \alpha_1(! |0\rangle \otimes \cdots \otimes ! |0\rangle) + \cdots + \alpha_{2^n}(! |1\rangle \otimes \cdots \otimes ! |1\rangle)} \quad Superpos$$

$$\frac{\alpha_r = 0, r \in \{1, \ldots, 2^n\} \qquad \Gamma \vdash \sum_{i=0}^{2^n} \alpha_i q_i}{\Gamma \vdash \sum_{\substack{i=1 \\ i \neq r}}^{2^n} \alpha_i q_i} \quad Simplif$$

Figura 4.2: Reglas para términos Bien-Formados del λ_q^M (continúa en la Figura 4.3).

Formalmente:

Definición 4.1 (Pre-término) *Llamamos* pre-término *a cualquier cadena* $\ddot{t}$ *generada por*

$$\frac{!x_1, \ldots, !x_n \vdash t}{!x_1, \ldots, !x_n \vdash !t} \quad \textit{Promotion}$$

$$\frac{\Gamma, x \vdash t}{\Gamma, !x \vdash t} \quad \textit{Dereliction}$$

$$\frac{\Gamma, !x, !y \vdash t}{\Gamma, !z \vdash t\ [z/x, z/y]} \quad \textit{Contraction}$$

$$\frac{\Gamma \vdash t}{\Gamma, !x \vdash t} \quad \textit{Weakening}$$

$$\frac{\Gamma, x \vdash t}{\Gamma, (\lambda x.t)} \quad \multimap\textit{-I}$$

$$\frac{\Gamma, !x \vdash t}{\Gamma \vdash (\lambda !x.t)} \quad \to\textit{-I}$$

$$\frac{\Gamma \vdash t_1 \qquad \Delta \vdash t_2}{\Gamma, \Delta \vdash (t_1\ t_2)} \quad \multimap\textit{-E}$$

Figura 4.3: Reglas para términos Bien-Formados del λ_q^M (continuación de la Figura 4.2).

la sintaxis descripta en la Figura 4.1.

Definición 4.2 (Término) *Un* término *es un pre-término* $\ddot{t}$ *tal que* $\vdash \ddot{t}$ *es derivable de las reglas que se presentan en las Figuras 4.2 y 4.3.*

La definición de valores, se extiende trivialmente como se muestra en la Figura 4.4

$$v ::= x \mid q \mid (\lambda x.t) \mid (\lambda !x.t) \mid !t \mid c_U \mid M_n$$

Figura 4.4: Definición de valores del λ_q^M.

4.3. Semántica Operacional

Para definir una semántica operacional que incluya a la medición, se debe tener en cuenta que dicha operación es intrínsecamente probabilística: dos aplicaciones de este operador sobre qubits que se encuentren exactamente en el mismo estado, pueden retornar diferentes resultados.

Alessandra Di Pierro, *et. al.*[12] han estudiado extensamente cómo definir un λ-cálculo probabilístico. En particular, su definición de transición probabilística (ver definición 11 de [12]) es lo que se necesita para el operador medición. La idea se resume con el siguiente ejemplo:

Ejemplo 4.3

$$\frac{\textit{Premisas sobre } P}{\begin{array}{c} P \rightarrow_p Q_1 \\ P \rightarrow_q Q_2 \end{array}}$$

Esta regla dice que si P es tal que cumple las premisas, P tiene probabilidad p de transicionar a Q_1 y probabilidad q de transicionar a Q_2.

Utilizando este concepto, se puede definir la regla de transición necesaria para la operación medición de la siguiente manera:

$$\text{M: } \frac{q = \alpha_1(!q_1) + \cdots + \alpha_{2^n}(!q_{2^n})}{\mathcal{H}; (M_{2^n}\ q) \rightarrow_{|\alpha_i|^2} !q_i}$$

Esta regla es la única que descarta el historial, ya que la medición es la única operación no reversible.

El resto de las reglas sufren algunos cambios:

- Debido a que se está trabajando con un λ-cálculo probabilístico, se debe asignar a todas las reglas que no involucren medición, probabilidad 1 de ocurrir.
- Se deben adaptar algunas reglas para la nueva definición de constantes.

El modelo operacional completo se presenta en la Figura 4.5.

La definición de Punto Fijo dado en la Sección 3.6 del Capítulo anterior sigue valiendo para el λ-cálculo con medición. Es fácil comprobar que bajo esa definición

$$fix\ !M_n \rightarrow_1 M_n!(fix\ !M_n).$$

Por supuesto, el teorema que nos permite razonar sin tener que acarear el historial, sigue valiendo con estas modificaciones.

Teorema 4.4 *En el cálculo cuántico con medición λ_q^M, la evolución del registro computacional está gobernada por las reglas de reducción que se listan en la Figura 4.6.* □

$$\text{APP}_1\text{:}\ \frac{t_1 \to_1 h_1; t_1'}{\mathcal{H}; (t_1\ t_2) \to_1 \mathcal{H}; (h_1\ \_); (t_1'\ t_2)}$$

$$\text{APP}_2\text{:}\ \frac{t_2 \to_1 h_2; t_2'}{\mathcal{H}; (v\ t_2) \to_1 \mathcal{H}; (\_\ h_2); (v\ t_2')}$$

$$\beta\text{:}\ \frac{}{\mathcal{H}; ((\lambda x.t)\ v) \to_1 \mathcal{H}; ((\lambda x.\bar{t}_x)\ \_); t\ [v/x]}$$

$$!\beta_1\text{:}\ \frac{}{\mathcal{H}; ((\lambda !x.t)\ !t') \to_1 \mathcal{H}; ((\lambda !x.\bar{t}_x)\ \_); t\ [t'/x]}\ \text{Si } x \in Free(t)$$

$$!\beta_2\text{:}\ \frac{}{\mathcal{H}; ((\lambda !x.t)\ !t') \to_1 \mathcal{H}; ((\lambda !x.\_)\ !t'); t}\ \text{Si } x \notin Free(t)$$

$$\text{U:}\ \frac{}{\mathcal{H}; (c_U\ \overline{q}) \to_1 \mathcal{H}; (c_U\ \_); U\ \overline{q}}\ \text{Donde } \overline{q} ::= q \mid !q$$

$$\text{M:}\ \frac{q = \alpha_1(!q_1) + \cdots + \alpha_{2^n}(!q_{2^n})}{\mathcal{H}; (M_{2^n}\ q) \to_{|\alpha_i|^2} !q_i}$$

$$\text{Id:}\ \frac{}{\mathcal{H}; t \to_1 \mathcal{H}; \_; t}\ \text{En otro caso}$$

Figura 4.5: Modelo operacional para el λ_q^M.

4.4. Ejemplos

4.4.1. Algoritmo de Deutsch

El algoritmo de Deutsch queda definido en λ_q^M de la siguiente manera:

$$\mathsf{deutsch}\ U_f \to_1 \ \ \textbf{let}\ x \otimes y = U_f\ ((H\ !|0\rangle) \otimes (H\ !|1\rangle))\ \textbf{in}\ M_1\ (H\ x)$$

Como se puede observar, no hay demasiado cambio en el algoritmo, ya que la medición se efectúa al final del mismo, pero el cambio importante es que aquí la medición es parte del lenguaje.

4.4.2. Teleportación

Este ejemplo es mucho más interesante, ya que en λ_q no quedaba claro el objetivo del algoritmo (ver Sección 3.8 del Capítulo anterior).

Aquí es donde se hace un uso real de la operación medición. En este caso, la medición está en medio del algoritmo, no al final. Luego de realizar dicha operación, se debe decidir

$$\text{APP}_1\text{:}\ \frac{t_1 \to_1 t_1'}{(t_1\ t_2) \to_1 (t_1'\ t_2)} \qquad \text{APP}_2\text{:}\ \frac{t_2 \to_1 t_2'}{(v\ t_2) \to_1 (v\ t_2')}$$

$$\beta\text{:}\ \frac{}{((\lambda x.t)\ v) \to_1 t\ [v/x]} \qquad !\beta_1\text{:}\ \frac{}{((\lambda !x.t)\ !t') \to_1 t\ [t'/x]}\ \text{Si } x \in Free(t)$$

$$!\beta_2\text{:}\ \frac{}{((\lambda !x.t)\ !t') \to_1 t}\ \text{Si } x \notin Free(t) \qquad \text{U:}\ \frac{}{(c_U\ \overline{q}) \to_1 U\ \overline{q}}\ \text{Donde } \overline{q} ::= q \mid !q$$

$$\text{M:}\ \frac{q = \alpha_0(!q_{00} \otimes \cdots \otimes !q_{10}) + \cdots + \alpha_{2^n}(!q_{02^n} \otimes \cdots \otimes !q_{n2^n})}{(M_{2^n}\ q) \to_{|\alpha_i|^2} (!q_{0i} \otimes \cdots \otimes !q_{1i})}$$

Figura 4.6: Evolución del registro computacional del λ_q^M.

si aplicar la compuerta Z y/o X en base al resultado. Para poder hacer esto, definimos la siguiente aplicación como sigue:

$$\mathsf{ex}\ x \to_1 !\,|0\rangle\ x^T + !\,|1\rangle\ (X\ x)^T$$

Como se puede comprobar

$$\mathsf{ex}\ !\,|1\rangle \to_1 X,$$
$$\mathsf{ex}\ !\,|0\rangle \to_1 I,$$

por lo tanto, $\mathsf{ex}\ b$ no es más que X^b.

También definimos

$$\mathsf{zed}\ x \to_1 Z\ (!\,|0\rangle\ (!\,|0\rangle)^T + x(!\,|1\rangle)^T) - !\,|0\rangle\ (!\,|1\rangle)^T + (X\ x)(!\,|1\rangle)^T$$

Análogamente al caso anterior

$$\mathsf{zed}\ !\,|1\rangle \to_1 Z,$$
$$\mathsf{zed}\ !\,|0\rangle \to_1 I,$$

por lo que $\mathsf{zed}\ b$ es Z^b.

Luego de estas dos definiciones, estamos en condiciones de definir la teleportación como sigue

$$\begin{aligned}\mathsf{teleport}\ q \to_1\ & \textbf{let}\ x \otimes y = \mathsf{epr}\ \textbf{in} \\ & \textbf{let}\ b_1 \otimes b_2 = M_2\ \mathsf{alice}\ (q, x)\ \textbf{in} \\ & \quad \mathsf{bob}\ (b_1, b_2, y)\end{aligned}$$

donde

$\mathsf{epr} \equiv cnot\ ((H\ !\,|0\rangle) \otimes !\,|0\rangle)$

$\mathsf{alice}\ (q, x) \rightarrow_1 \ \mathbf{let}\ r \otimes w = ((cnot\ q) \otimes x)\ \mathbf{in}$
$((H\ r) \otimes w)$

$\mathsf{bob}\ (b_1, b_2, y) \rightarrow_1 (\mathsf{zed}\ b_1)\ (\mathsf{ex}\ b_2)\ y$

Notemos que aquí se ve claramente que luego de las operaciones realizadas por Alice, se realiza una medición, por lo que tanto b_1 como b_2 pueden codificarse como bits clásicos para pasarlos como argumentos a Bob.

4.5. Resumen del Capítulo

En este capítulo se ha extendido el λ-Cálculo Cuántico de André van Tonder para que la medición sea parte del cálculo. Además, se han mostrado algunos ejemplos de uso, incluido el algoritmo de Teleportación, tal como se discutió en la Sección 3.8.

Para ello se han realizado varios cambios al modelo. Se separó la definición de constantes en constantes para compuertas, constantes para qubits y una constante aparte para la medición. De esta manera, se le pudo dar una sintaxis definida a los qubits, a fin de que sirva para realizar la medición.

La regla de transición para la medición hace uso del modelo de reglas probabilísticas de Di Pierro *et. al.*[12].

En el siguiente Capítulo se plantearán las conclusiones y algunos interrogantes que han quedado abiertos para trabajos futuros.

Capítulo 5

Conclusiones y trabajo futuro

El λ-cálculo cuántico permite razonar con los algoritmos de una manera más familiar que los circuitos cuánticos, además, provee una herramienta para estudiar la semántica de las operaciones.

Una de las consecuencias de tener la operación medición como parte del cálculo es que no se puede desarrollar una teoría ecuacional completa. Si bien la teoría desarrollada por van Tonder sigue siendo válida, no puede considerarse completa sin la medición. Qué consecuencias trae aparejado este hecho, es tema de trabajo futuro: *La computación cuántica no se puede plantear como una teoría ecuacional.* Por supuesto, si pensamos en experimentos físicos, donde la medición es probabilística, queda muy claro que no pueda realizarse una teoría ecuacional, pero desde el punto de vista computacional, habrá que hacer un estudio más profundo de ello.

van Tonder demuestra en su primer trabajo[28] sobre el λ-cálculo cuántico que su modelo es equivalente a la Máquina de Turing Cuántica[11], por lo tanto, agregar medición al lenguaje no amplia el modelo computacional, pero provee una manera más intuitiva de razonar con el lenguaje (ver Sección 3.8).

En un trabajo más reciente, André van Tonder junto a Miquel Dorca[29], definieron una teoría de tipos y dieron una Semántica Categórica para el λ_q. Queda por extender dichos resultados al cálculo con medición aquí desarrollado. En contraposición, una teoría de tipos ya ha sido desarrollada por Peter Selinger y Benoît Valiron[25], pero su trabajo es completamente diferente del nuestro, ya que en su definición del λ-cálculo, llevan un control clásico, no probabilístico.

Bibliografía

[1] S. Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111(1–2):3–57, 1993.

[2] T. Altenkirch and J. J. Grattage. A functional quantum programming language. In *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS)*. IEEE Computer Society, 2005.

[3] T. Altenkirch and J. J. Grattage. QML: Quantum data and control. Submitted for publication, 2005.

[4] T. Altenkirch, J. J. Grattage, J. K. Vizzotto, and A. Sabry. An algebra of pure quantum programming. *Electronic Notes in Theoretical Computer Science*, 170:23–47, 2007.

[5] P. Arrighi and G. Dowek. Operational semantics for a formal tensorial calculus. In *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004.

[6] P. Arrighi and G. Dowek. Linear-algebraic λ-calculus. `arXiv:quant-ph/0501150`, 2005.

[7] G. D. Baker. Qgol: a system for simulating quantum computations: theory, implementation and insight. Honours thesis, Macquarie University, 1996.

[8] C. H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, 1973.

[9] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters. Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Phys. Rev. Lett.*, 70(13):1895–1899, 1993.

[10] J. Brown. *The Quest for the Quantum Computer*. Touchtone, New York, NY, USA, 2001.

[11] D. Deutsch. Quantum theory, the church-turing principle, and the universal quantum computer. In *Proceedings of the Royal Society of London*, volume A400, pages 97–117, 1985.

[12] A. Di Pierro, C. Hankin, and H. Wiklicky. Probabilistic λ-calculus and quantitative program analysis. *Journal of Logic and Computation*, 15(2):159–179, 2005.

[13] A. Einstein, B. Podolsky, and N. Rosen. Can quantum-mechanical description of physical reality be considered complete? *Phys. Rev.*, 47(10):777–780, 1935.

[14] R. Feynman, R. Leighton, and M. Sands. *The Feynman Lectures on Physics*, volume III. Addison-Wesley, 1965.

[15] S. J. Gay. Quantum programming languages: survey and bibliography. *Mathematical Structures in Computer Science*, 16(4):581–600, 2006.

[16] J. J. Grattage. *QML: A functional quantum programming language*. PhD thesis, University of Nottingham, 2006.

[17] L. K. Grover. A fast quantum mechanical algorithm for database search. In *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery.

[18] E. H. Knill. Conventions for quantum pseudocode. Technical Report LAUR-96-2724, Los Alamos National Laboratory, 1996.

[19] J. Maraist, M. Odersky, D. Turner, and P. Wadler. Call-by-name call-by-value, call-by-need and the linear lambda calculus. *Theoretical Computer Science*, 228(1-2):175–210, 1999.

[20] P. Maymin. Extending the lambda calculus to express randomized and quantumized algorithms. `arXiv:quant-ph/961252`, 1996.

[21] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press., 2000.

[22] R. A. G. Seely. Linear logic, $*$-autonomous categories and cofree coalgebras. In *Categories in Computer Science and Logic*, volume 92, pages 371–382, Providence, Rhode Island, 1989. American Mathematical Society.

[23] P. Selinger. A brief survey of quantum programming languages. In *Functional and Logic Programming*, volume 2998 of *Lecture Notes in Computer Science*, pages 1–6. Springer Berlin / Heidelberg, 2004.

[24] P. Selinger. Towards a quantum programming language. *Mathematical Structures in Computer Science*, 14(4):527–586, 2004.

[25] P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. *Mathematical Structures in Computer Science*, 16(3):527–552, 2006.

[26] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[27] B. Valiron. A functional programming language for quantum computation with classical control. Master's thesis, University of Ottawa, 2004.

[28] A. van Tonder. A lambda calculus for quantum computation. *SIAM Journal on Computing*, 33(5):1109–1135, 2004.

[29] A. van Tonder and M. Dorca. Quantum computation, categorical semantics and linear logic. `arXiv:quant-ph/0312174v4`, 2007.

[30] P. Wadler. A syntax for linear logic. In *Proceedings of the 9th International Conference on Mathematical Foundations of Programming Semantics*, pages 513–529, London, UK, 1994. Springer-Verlag.

[31] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.