λ-Cálculo Cuántico

(de André van Tonder [vT04])

Alejandro Díaz-Caro

Departamento de Ciencias de la Computación
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Universidad Nacional de Rosario

16 de diciembre de 2007

Contenido de la presentación

Motivación

Actualmente existen dos modelos equivalentes[Yao93] predominantes para "pensar" la computación cuántica:

- Máquinas de Turing Cuánticas[Ben80][Deu85]
- Circuitos Cuánticos[Deu89]

Las Máquinas de Turing Cuánticas proveen un modelo para definir la universalidad de la Computación Cuántica, pero razonar sobre ellas es un proceso bastante complicado. Por ese motivo los circuitos cuánticos son más populares: proveen una visión gráfica y composicional de los algoritmos y pueden ser manipulados algebraicamente. Pero ningún circuito cuántico finito puede ser universal.

En Computación Clásica, el λ-Cálculo provee un modelo alternativo, equivalente a las Máquinas de Turing, y es de gran utilidad en la teoría de la computación y el estudio de lenguajes y sus semánticas.

La idea es proveer a la Computación Cuántica de una herramienta equivalente.

Computación Cuántica (Physics-free)

Qubits

La Computación Cuántica es un modelo de computación basado en la Mecánica Cuántica.
Su unidad básica es el "Qubit"

Definición

Un qubit es un vector de dos dimensiones de la siguiente forma

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

con $\alpha, \beta \in \mathbb{C}$ y $|\alpha|^2 + |\beta|^2 = 1$. Esto forma un Espacio Vectorial

Una base del espacio vectorial de un qubit es

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\}$$

A estos vectores los notamos de la siguiente manera

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

Entonces, un qubit queda definido por

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

Para un sistema de n-qubits tendremos el espacio vectorial de dimensión 2^n generado por la base

$$\left\{ \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right\}$$

y a esos n-qubits los notamos de dos maneras alternativas:

$$|i_1, \ldots, i_n\rangle \text{ con } i_k \in \{0, 1\}$$
$$\text{o}$$
$$|i\rangle \text{ con } i \in \{0, \ldots 2^{n-1}\}$$

Entonces un n-qubit queda definido por

$$\sum_{k=0}^{2^{n-1}} \alpha_k |k\rangle \text{ tal que } \sum_{k=0}^{2^{n-1}} |\alpha_k|^2 = 1$$

Computación Cuántica (Physics-free)

Compuertas

Las compuertas cuánticas son matrices que satisfacen determinadas propiedades (básicamente, propiedades que hacen que la matriz "aplicada a" (multiplicada por) los qubits nos devuelvan qubits).

Ejemplo

Compuerta NOT

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$X |0\rangle = X \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$
$$X |1\rangle = X \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

Ejemplo

Compuerta Hadamard

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

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

Ejemplo

Compuerta CNOT

$$CNOT = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}$$

$$CNOT\,|00\rangle = |00\rangle$$
$$CNOT\,|01\rangle = |01\rangle$$
$$CNOT\,|10\rangle = |11\rangle$$
$$CNOT\,|11\rangle = |10\rangle$$

La aplicación de una compuerta cuántica a cualquier qubit es una aplicación lineal

$$U\sum_{i=0}^{2^n-1}\alpha_i\,|i\rangle = \sum_{i=0}^{2^n-1}\alpha_i U\,|i\rangle$$

Por lo tanto, especificando cómo actúa la compuerta en la base del espacio de qubits, ya se ha especificado todo lo necesario.

Ejemplo

$$Z\,|0\rangle = |0\rangle$$
$$Z\,|1\rangle = -\,|1\rangle$$

Entonces
$Z(\alpha\,|0\rangle + \beta\,|1\rangle) = \alpha\,|0\rangle - \beta\,|1\rangle$

Obs Todas las compuertas cuánticas son reversibles y coinciden con su inversa, entonces, aplicando dos veces una compuerta, se vuelve al estado original.

Computación Cuántica (Physics-free)

Medición

Al medir un n-qubits $\sum_{i=0}^{2^n-1}\alpha_i\,|i\rangle$ respecto a la base $\{|i\rangle\}_{i=0}^{2^n-1}$ del espacio de n-qubits, obtendré un vector $|k\rangle$ de dicha base con probabilidad $|\alpha_k|^2$.

Ejemplo

Al medir el qubit $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ obtendré $|0\rangle$ ó $|1\rangle$ con probabilidad $\frac{1}{2}$ cada uno.

Obs Un algoritmo cuántico se basa en hacer evolucionar un sistema de n-qubits mediante la aplicación de compuertas y mediciones. Debido a la reversibilidad de las compuertas, los algoritmos son reversibles (excepto en la medición).

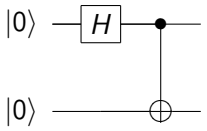
Computación Cuántica (Physics-free)

Circuitos

Un circuito cuántico es la representación gráfica de un algoritmo cuántico.

Ejemplo

Dado $|00\rangle$, aplicar H al primer qubit y luego un $CNOT$ entre el primero y el segundo.



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

Computación Cuántica (Physics-free)

Entanglement

En el ejemplo anterior, hemos producido un estado entangled, los cuales son estados en que *no puedo separar los qubits*.

Ejemplo

No entangled

$$\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$

$$= \underbrace{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)}_{\text{1er qubit}} \underbrace{|0\rangle}_{\text{2do qubit}}$$

Ejemplo

Entangled

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

No los puedo separar!

Primer Intento

Presentación

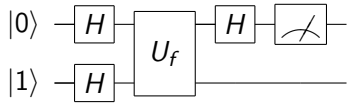
- En lambda cálculo *clásico*, las β-reducciones descartan información en cada paso, haciendo el proceso irreversible.
- Cualquier cómputo clásico se puede transformar en un cómputo reversible[Ben73]

Podríamos tener reversibilidad de la siguiente manera:
Sea x un término y $\beta : x \to \beta(x)$ una β-reducción. Entonces consideremos la función $x \to (x, \beta(x))$, la cual es invertible.

En una versión simple, el cómputo procede de la siguiente manera $x \to (x, \beta(x)) \to (x, \beta(x), \beta^2(x)) \to (x, \beta(x), \beta^2(x), \beta^3(x)) \to \ldots$
Aunque éste método podría funcionar para implementar reversibilidad, veremos que no funciona en el caso cuántico sin hacerle algunas modificaciones.

Continuando con la Motivación...

Circuito cuántico



En Lambda Cálculo Cuántico se podría escribir así:

Deutsch

$$\textit{deutsch}\ U_f \equiv \textit{let}\ (x, y) = U_f\ ((H\ 0), (H\ 1))\ \textit{in}\ ((H\ x), y)$$

Sintaxis para el primer intento

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

0 y 1 son primitivas.
El resto de las constantes denotan compuertas elementales entre qubits.

Ahora le permitimos al estado de un cómputo ser una superposición cuántica de términos en este lenguaje.
Consideremos un estado inicial como el siguiente string: $|(H\ 0)\rangle$
Quisiéramos elegir reglas de transición tales que este estado evalúe una compuerta Hadamard aplicada a $|0\rangle$.

Una regla de transición candidata podría ser:

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

Y usamos el truco para hacerlo reversible:

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

El ";" denota la concatenación de strings. El resultado se ha factorizado a la derecha.

Segundo intento

Presentación

Con sólo guardar en cada paso qué subtérmino se ha reducido y qué operación se ha aplicado ya me bastaría.

Ejemplo

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

En cada paso reemplazamos los subtérminos que no necesitamos para la reversibilidad por el placeholder "_".

Ahora formalicemos un poco ésto.

Primer Intento

Problemas

Ejemplo

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

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

Aquí no puedo factorizar el resultado ya que quedó en entangled con el término intermedio del historial
Este método está guardando más información de la necesaria para lograr reversibilidad.

Primero extendemos la definición de valores para incluir a las constantes

Definición de valores del Segundo Intento

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

El estado computacional se toma como una superposición cuántica de secuencias de la forma

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

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

Las reglas de transición son las siguientes

Reglas de transición del Segundo Intento

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

$$\frac{t_2 \to h_2; t_2'}{\mathcal{H}; (v_1\ t_2) \to \mathcal{H}; (\_\ h_2); (v_1\ t_2')}\ (\text{APP}_2)$$

$\mathcal{H}$ denota el historial (puede estar vacío)

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

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

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

(Ver formalización en anexo)

$\bar{t}_x$ se obtiene de t reemplazando recursivamente todos los subtérminos que no contienen x con el símbolo _ y manteniendo x.

Ejemplo

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

En este ejemplo podemos usar como criterio de terminación comparar la última expresión con "_", no tenemos un criterio de terminación bien definido en λ_i ya que el estado podría tener una superposición de varios historiales, algunos de los cuales hayan terminado y otros no. Entonces "observando" podría cambiar el estado.
Este problema será resuelto más adelante con el λ_q.

Regla de reducción extra para símbolos de compuertas cuánticas

Regla de reducción para símbolos de compuertas cuánticas

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

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

Ejemplo

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

Segundo Intento

Problemas

Ejemplo

$|((\lambda x.cosa)\ otraCosa)\rangle \to |((\lambda x.\_)\ otraCosa); cosa\rangle$
Aquí debemos guardar "*otraCosa*" en el historial para mantener reversibilidad.

Pero entonces entramos en problemas cuando el argumento que se descarta es una superposición

Ejemplo

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

Aquí tengo dos formas de reducir $(\lambda x.0)\ (\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))$

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

Otro ejemplo

Ejemplo

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

Quedó el historial en entangled con el estado!!!

Tercer (y último) intento

λ-Cálculo Cuántico (λ_q)

Definición

Decimos que una subexpresión es definida con respecto a la base computacional si es textualmente la misma en todos los branches de la superposición.

Ejemplo

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

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

Para resolver los problemas que mostramos previamente, consideraremos 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 al tipo de elementos llamado λ-cálculo lineal, que ha sido estudiado extensamente [Abr93][Wad94][MOTW99][See89]

La sintaxis que usaremos será una extensión de la introducida en [Wad94]

Sintaxis del λ_q

$t ::=$		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

Observaciones:

- Los términos de la forma $!t$ son llamados no lineales. Los términos no lineales serán 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.
- En una abstracción lineal puedo usar todos los términos no lineales que quiera (o ninguno), pero debe haber un término lineal (y sólo uno) en el cuerpo de la función.

Para reforzar estas reglas, necesitamos términos "bien-formados". 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

Bien-Formados	No bien-formados
$(\lambda !x.0)$	$(\lambda x.0)$
$(\lambda x.x)$	$(\lambda x.!x)$
$(\lambda !x.(x\ x))$	$(\lambda x.(x\ x))$
$(\lambda y.(\lambda !x.y))$	$(\lambda y.(\lambda x.y))$
$(\lambda !y.!(\lambda !x.y))$	$(\lambda y.!(\lambda !x.y))$

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.

Ejemplo

La expresión $((\lambda !x.0)\ !(H\ 0))$ es bien-formada. El problema está en que permitimos usar ! para promover la expresión $(H\ 0)$ (que va a ser descartada) a un valor no lineal. Si reducimos primero el subtérmino $(H\ 0)$, razonando ecuacionalmente tenemos
$|((\lambda !x.0)\ !(H\ 0)\rangle = \frac{1}{\sqrt{2}}(|((\lambda !x.0)\ !0)\rangle + |((\lambda !x.0)\ !1)\rangle) = \sqrt{2}\,|0\rangle$
En cambio, si consideramos a $!(H\ 0)$ como irreducible, podríamos usar beta reducción inmediatamente y obtener
$|((\lambda !x.0)\ !(H\ 0))\rangle = |0\rangle$

Para prevenir que términos de la forma $!t$ sean evaluados, seguimos a [Abr93] y extendemos nuestra definición de valores de la siguiente manera

Valores en el λ_q

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

1. De acuerdo a estas reglas, las superposiciones cuánticas sólo pueden ser creadas mediante la evaluación de términos que contengan primitivas cuánticas.
2. El resultado de aplicar una compuerta cuántica es un valor lineal (sin !).
3. Nótese que cuando una función no lineal encuentra un término lineal, lo único que se puede aplicar es la regla (Id).
4. Las reglas de reducción precedentes permiten crear superposición, pero los términos en la superposición sólo difieren en las posiciones que contienen las constantes 0 y 1. A continuación vamos a formalizar esto último.

El modelo computacional se describe como sigue (donde $\bar{t}$ es definido como lo definimos anteriormente).

Modelo operacional para el λ_q

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

$$\frac{t_2 \to h_2; t_2'}{\mathcal{H}; (v\ t_2) \to \mathcal{H}; (\_\ h_2); (v_1\ t_2')}\ (\text{APP}_2)$$

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

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

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

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

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

Definición

Dos términos son congruentes si coinciden símbolo a símbolo excepto tal vez en las posiciones que contienen 0 ó 1.

Lema

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

Lema

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 siguente lema:

Lema

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

Observación: Notar que mediante reducción tampoco puedo crear !-suspenciones 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.

Teorema

En el cálculo cuántico λ_q, la evolución del registro computacional está gobernada por las reglas de reducción que se listan a continuación.

Reglas de reducción del registro computacional

$$\frac{t_1 \to t_1'}{(t_1\ t_2) \to (t_1'\ t_2)}\ (\text{APP}_1) \qquad \frac{}{(\lambda!x.t)\ !t' \to t[t'/x]}\ (!\beta_1)\ \text{Si } x \in F(t)$$

$$\frac{t_2 \to t_2'}{(v_1\ t_2) \to (v_1\ t_2')}\ (\text{APP}_2) \qquad \frac{}{(\lambda!x.t)\ !t' \to t}\ (!\beta_2)\ \text{Si } x \notin F(t)$$

$$\frac{}{(\lambda x.t)\ v \to t[v/x]}\ (\beta) \qquad \frac{}{|c_U\ \phi\rangle \to U\,|\phi\rangle}\ (\text{U})$$

□

Observación: Llegamos a un conjunto de reglas de reducción simple y que nos permite razonar sobre los cómputos sin tener que acarrear el historial.

Lema

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

Corolario

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. □

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

Ahora, para presentar la teoría ecuacional, listamos el conjunto de axiomas y reglas de inferencia de dicha teoría.

Sistema de prueba ecuacional para el cálculo cuántico λ_q

$$\frac{}{t = t}\ (\text{refl}) \qquad \frac{t_1 = t_2}{\lambda!x.t_1 = \lambda!x.t_2}\ (\lambda_2)$$

$$\frac{t_1 = t_2}{t_2 = t_1}\ (\text{sim}) \qquad \frac{}{(\lambda x.t)\ v = t[v/x]}\ (\beta)$$

$$\frac{t_1 = t_2 \quad t_2 = t_3}{t_1 = t_3}\ (\text{trans}) \qquad \frac{}{(\lambda!x.t)\ !t' = t[t'/x]}\ (!\beta_1)\ \text{Si } x \in F(t)$$

$$\frac{t_1 = t_2 \quad t_3 = t_4}{(t_1\ t_3) = (t_2\ t_4)}\ (\text{app}) \qquad \frac{}{(\lambda!x.t)\ !t' = t}\ (!\beta_2)\ \text{Si } x \notin F(t)$$

$$\frac{t_1 = t_2}{\lambda x.t_1 = \lambda x.t_2}\ (\lambda_1) \qquad \frac{}{|c_U\ \phi\rangle = U\,|\phi\rangle}\ (\text{U})$$

Notar que no hay ninguna regla que permita sustituciones dentro de las !-suspenciones.

Teorema

En el Lambda Cálculo Cuántico, la evolución del registro computacional procede reemplazando términos por términos iguales de acuerdo a la teoría ecuacional anterior. □

Cómo computa el λ_q

Punto Fijo

Vamos a definir el operador de punto fijo en λ_q

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

$$(v\ !v)$$

Veamos cómo actúa

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

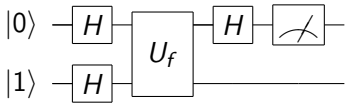
Ejemplo

Si $t \equiv \lambda!f.u$
$fix\ !t \rightarrow t\ !(fix\ !t) \equiv (\lambda!f.u)\ !(fix\ !t)$
$\rightarrow u[(fix\ !t)/f]$
En otras palabras, $fix\ !t$ se copia a sí mismo en el cuerpo (u) de t a traves de la reducción.

Ejemplos

Un algoritmo muy conocido en Computación Cuántica es el llamado "Algoritmo de Deutsch"[NC00].

Circuito cuántico



Donde U_f es una compuerta que actúa de la siguiente manera

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

donde f es alguna función desconocida de 1 bit.

La salida de este circuito será

$$\begin{cases} \pm\,|0\rangle\,|+\rangle & \text{si } f(0) = f(1) \\ \pm\,|1\rangle\,|+\rangle & \text{si } f(0) \neq f(1) \end{cases}$$

(Ver demostración en anexo)
Por lo tanto, midiendo el primer qubit se puede saber si f es constante o no.

En Lambda Cálculo Cuántico se podría escribir así:

Deutsch

$$deutsch\ U_f \equiv let\ (x,y) = U_f\ ((H\ 0),(H\ 1))\ in\ ((H\ x),y)$$

Ejemplo

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

$$U_f\ |00\rangle \to |00\rangle$$
$$U_f\ |01\rangle \to |01\rangle$$
$$U_f\ |10\rangle \to |11\rangle$$
$$U_f\ |11\rangle \to |10\rangle$$
$$\therefore U_f \equiv cnot$$

Muchas gracias
(a quienes no se fueron a mitad de la charla)
FIN

Ejemplo

Entonces

$$deutsch\ cnot \equiv let\ (x,y) = cnot\ ((H\ 0),(H\ 1))\ in\ ((H\ x),y)$$

Desarrollemos $cnot\ ((H\ 0),(H\ 1))$
$cnot\ ((H\ 0),(H\ 1)) = cnot\ (\frac{1}{2}(|0\rangle + |1\rangle), \frac{1}{2}(|0\rangle - |1\rangle))$
$= cnot\ (\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle))$
$= \frac{1}{2}(|00\rangle - |01\rangle + |11\rangle - |10\rangle) = (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle))$
Volviendo: $let\ (x,y) = (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle))\ in$
$((H\ x),y) = (H\ (\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle))$
$= |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Bibliografía I


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


Paul Benioff.
The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines.
Journal of Statistical Physics, V22(5):563–591, May 1980.


David Deutsch.
Quantum theory, the church-turing principle and the universal quantum computer.
Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 400(1818):97–117, 1985.

Bibliografía II

David Deutsch.
Quantum computational networks.
Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 425(1868):73–90, 1989.

John Maraist, Martin Odersky, DavidÑ. Turner, and Philip Wadler.
Call-by-name call-by-value, call-by-need and the linear lambda calculus.
Theoretical Computer Science, 228(1-2):175–210, 1999.

Michael Nielsen and Isaac Chuang.
Quantum Computation and Quantum Information.
Cambridge University Press, October 2000.

Bibliografía III

Robert A. G. Seely.
Linear logic, $*$-autonomous categories and cofree coalgebras.
In John W. Gray and Andre Scedrov, editors, *Categories in Computer Science and Logic*, volume 92, pages 371–382, Providence, Rhode Island, 1989. American Mathematical Society.

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

Philip 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.

Bibliografía IV

Andrew Yao.
Quantum circuit complexity.
In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 352–361, Los Alamitos, CA, 1993. Institute of Electrical and Electronic Engineers Computer Society Press.

Charles H. Bennet.
Logical reversibility of computation.
IBM Journal of Research and Development, 17(525532), 1973

Anexo

Desarrollo del Algoritmo de Deutsch

$$
\begin{aligned}
|01\rangle &\xrightarrow{H(1,2)} \tfrac{1}{2}(|0\rangle+|1\rangle)(|0\rangle-|1\rangle) = \tfrac{1}{2}(|00\rangle-|01\rangle+|10\rangle-|11\rangle)\\
&\xrightarrow{U_f} \tfrac{1}{2}(U_f|00\rangle-U_f|01\rangle+U_f|10\rangle-U_f|11\rangle)\\
&= \tfrac{1}{2}(|0,f(0)\rangle-|0,\neg f(0)\rangle+|1,f(1)\rangle-|1,\neg f(1)\rangle)\\
&= \tfrac{1}{2}(|0\rangle(|f(0)\rangle-|\neg f(0)\rangle)+|1\rangle(|f(1)\rangle-|\neg f(1)\rangle))\\
&\xrightarrow{H(1)} \tfrac{1}{2\sqrt{2}}((|0\rangle+|1\rangle)(|f(0)\rangle-|\neg f(0)\rangle)+(|0\rangle-|1\rangle)(|f(1)\rangle-|\neg f(1)\rangle))
\end{aligned}
$$

Si $f(0)=f(1)$

$$
\begin{aligned}
&= \tfrac{1}{2\sqrt{2}}(|0\rangle(|f(0)\rangle-|\neg f(0)\rangle+|f(0)\rangle-|\neg f(0)\rangle)+|1\rangle(|f(0)\rangle-|\neg f(0)\rangle-|f(0)\rangle+|\neg f(0)\rangle))\\
&= \tfrac{1}{2\sqrt{2}}(|0\rangle(2|f(0)\rangle-2|\neg f(0)\rangle))\\
&= |0\rangle(\pm\tfrac{1}{\sqrt{2}}(|0\rangle+|1\rangle))
\end{aligned}
$$

Si $f(0)\neq f(1)$

$$
\begin{aligned}
&= \tfrac{1}{2\sqrt{2}}(|0\rangle(|f(0)\rangle-|\neg f(0)\rangle-|f(0)\rangle+|\neg f(0)\rangle)+|1\rangle(|f(0)\rangle-|\neg f(0)\rangle+|f(0)\rangle-|\neg f(0)\rangle))\\
&= \tfrac{1}{2\sqrt{2}}(|1\rangle(2|f(0)\rangle-2|\neg f(0)\rangle))\\
&= |1\rangle(\pm\tfrac{1}{\sqrt{2}}(|0\rangle+|1\rangle))
\end{aligned}
$$

Anexo

Formalización del $\overline{t}_x$

Definición

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