Fundamentos de lenguajes de programación cuántica

Día 4:
$\sim$ Paradigma del "co-procesador cuántico": Control clásico y datos cuánticos $\backsim$

Alejandro Díaz-Caro

Universidad Nacional de Quilmes

22º Escuela de Verano de Ciencias Informáticas

Río Cuarto, Córdoba – 9 al 14 de febrero de 2015

Historia

André van Tonder

SIAM Journal on Computing 33(5), 1109–1135, 2004

Primera extensión de lambda cálculo para computación cuántica

Peter Selinger

Mathematical Structures in Computer Science 14(4), 527–586, 2004

"Clasical Control, Quantum Data"

- Flujo de control clásico que controla una máquina cuántica
- Indica qué operaciones aplicar

Peter Selinger y Benoît Valiron

Mathematical Structures in Computer Science 16(3), 527–552, 2006

PhD Thesis de Valiron, Ottawa, 2008

Extensión cuántica a lambda cálculo con control clásico

Alexander S. Green, Peter LeFanu Lumsdaine, Neil J. Ross, Peter Selinger y Benoît Valiron

34th Conference on Programming Language Design and Implementations, 2013

Un lenguaje de programación cuántica escalable

`http://www.mathstat.dal.ca/~selinger/quipper`

El cálculo de van Tonder

Premisas

- No clonado $\Rightarrow$ lógica lineal (más en Selinger–Valiron)

El cálculo de van Tonder

Premisas

- No clonado $\Rightarrow$ lógica lineal (más en Selinger–Valiron)
- Para toda compuerta U, $UU = I$ $\therefore$ cc. es reversible

El cálculo de van Tonder

Premisas

- No clonado $\Rightarrow$ lógica lineal (más en Selinger–Valiron)
- Para toda compuerta U, $UU = I$ $\therefore$ cc. es reversible
- Todo computo se puede hacer reversible (C. Bennett 1973)

El cálculo de van Tonder

Premisas

- No clonado $\Rightarrow$ lógica lineal (más en Selinger–Valiron)
- Para toda compuerta U, $UU = I$ $\therefore$ cc. es reversible
- Todo computo se puede hacer reversible (C. Bennett 1973)

Ejemplo Dado un término $\mathbf{t}$, $\beta(\mathbf{t})$ es su reducto en un paso

$$(\mathbf{t}) \to (\mathbf{t}, \beta(\mathbf{t})) \to (\mathbf{t}, \beta(\mathbf{t}), \beta^2(\mathbf{t})) \to \cdots$$

El cálculo de van Tonder

Premisas

- No clonado $\Rightarrow$ lógica lineal (más en Selinger–Valiron)
- Para toda compuerta U, $UU = I$ $\therefore$ cc. es reversible
- Todo computo se puede hacer reversible (C. Bennett 1973)

Ejemplo Dado un término $\mathbf{t}$, $\beta(\mathbf{t})$ es su reducto en un paso

$$(\mathbf{t}) \to (\mathbf{t}, \beta(\mathbf{t})) \to (\mathbf{t}, \beta(\mathbf{t}), \beta^2(\mathbf{t})) \to \cdots$$

Principal idea del cálculo de van Tonder
Llevar un historial para hacer el cálculo reversible

El cálculo

$$\begin{array}{rcl} \mathbf{t}, \mathbf{r} & ::= & x \mid \lambda x.\mathbf{t} \mid \mathbf{t}\ \mathbf{r} \mid c \\ c & ::= & 0 \mid 1 \mid H \mid CNOT \mid X \mid Z \mid \cdots \end{array}$$

Reglas de reducción

Ejemplo: $|H0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

El cálculo

$$\begin{array}{rcl} \mathbf{t}, \mathbf{r} & ::= & x \mid \lambda x.\mathbf{t} \mid \mathbf{t}\ \mathbf{r} \mid c \\ c & ::= & 0 \mid 1 \mid H \mid CNOT \mid X \mid Z \mid \cdots \end{array}$$

Reglas de reducción

Ejemplo: $|H0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

Pero con un historial...

$|H0\rangle \rightarrow \frac{1}{\sqrt{2}}(|(H0); 0\rangle + |(H1); 1\rangle)$

El problema

$$
\begin{aligned}
|H(H0)\rangle \rightarrow & \frac{1}{\sqrt{2}}|H(H0);(H0)\rangle + |H(H0);(H1)\rangle \\
= & |H(H0)\rangle \otimes \frac{1}{\sqrt{2}}|H0\rangle + |H1\rangle \\
\rightarrow & \frac{1}{2}|H(H0)\rangle \otimes |H0,0\rangle + |H0;1\rangle + |H1;0\rangle - |H1;1\rangle
\end{aligned}
$$

El problema

$$
\begin{aligned}
|H(H0)\rangle \rightarrow & \frac{1}{\sqrt{2}}|H(H0);(H0)\rangle + |H(H0);(H1)\rangle \\
= & |H(H0)\rangle \otimes \frac{1}{\sqrt{2}}|H0\rangle + |H1\rangle \\
\rightarrow & \frac{1}{2}|H(H0)\rangle \otimes \underbrace{|H0,0\rangle + |H0;1\rangle + |H1;0\rangle - |H1;1\rangle}_{\textbf{¡Historial y registro enredados!}}
\end{aligned}
$$

El problema

$$
\begin{aligned}
|H(H0)\rangle \rightarrow & \frac{1}{\sqrt{2}}|H(H0);(H0)\rangle + |H(H0);(H1)\rangle \\
= & |H(H0)\rangle \otimes \frac{1}{\sqrt{2}}|H0\rangle + |H1\rangle \\
\rightarrow & \frac{1}{2}|H(H0)\rangle \otimes \underbrace{|H0,0\rangle + |H0;1\rangle + |H1;0\rangle - |H1;1\rangle}_{\text{¡Historial y registro enredados!}}
\end{aligned}
$$

Solución No guardar TANTA información en el historial

$$
\begin{aligned}
|H(H0)\rangle \rightarrow & \frac{1}{\sqrt{2}}|\_(H\_);(H0)\rangle + |\_(H\_);(H1)\rangle \\
= & |\_(H\_)\rangle \otimes \frac{1}{\sqrt{2}}|H0\rangle + |H1\rangle \\
\rightarrow & \frac{1}{2}|\_(H\_)\rangle \otimes |H\_,0\rangle + |H\_;1\rangle + |H\_;0\rangle - |H\_;1\rangle \\
= & |\_(H\_);H\_\rangle \otimes |0\rangle
\end{aligned}
$$

Ejemplos (en lugar de dar las reglas de reducción)

$$|(\lambda x.\mathbf{t})\ \mathbf{r}\rangle \rightarrow |(\lambda x.\_)\ \mathbf{r}; \mathbf{t}\rangle$$

Guardamos **r** porque de otra manera se pierde

Ejemplos (en lugar de dar las reglas de reducción)

$$|(\lambda x.\mathbf{t})\ \mathbf{r}\rangle \rightarrow |(\lambda x.\_)\ \mathbf{r};\mathbf{t}\rangle$$

Guardamos **r** porque de otra manera se pierde

$$|(\lambda x,0)\ (H0)\rangle \rightarrow |\_(H\_)\rangle \otimes \left|(\lambda x,0)(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))\right\rangle$$

Dos formas de reducir:

$$|\_(H\_);(\lambda x.\_)|+\rangle\rangle \otimes |0\rangle$$

Ejemplos (en lugar de dar las reglas de reducción)

$$|(\lambda x.\mathbf{t})\ \mathbf{r}\rangle \to |(\lambda x.\_)\ \mathbf{r};\mathbf{t}\rangle$$

Guardamos **r** porque de otra manera se pierde

$$|(\lambda x,0)\ (H0)\rangle \to |\_(H\_)\rangle \otimes \left|(\lambda x,0)(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))\right\rangle$$

Dos formas de reducir:

$$|\_(H\_);(\lambda x.\_)|+\rangle\rangle \otimes |0\rangle$$

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

¡El segundo caso no está normalizado!

Ejemplos (en lugar de dar las reglas de reducción)

$$|(\lambda x.\mathbf{t})\ \mathbf{r}\rangle \to |(\lambda x.\_)\ \mathbf{r};\mathbf{t}\rangle$$

Guardamos **r** porque de otra manera se pierde

$$|(\lambda x,0)\ (H0)\rangle \to |\_(H\_)\rangle \otimes \left|(\lambda x,0)(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))\right\rangle$$

Dos formas de reducir:

$$|\_(H\_);(\lambda x.\_)|+\rangle\rangle \otimes |0\rangle$$

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

¡El segundo caso no está normalizado!

Muchos problemas más, que se solucionan con un sistema de reescritura meticulosamente elegido teniendo en cuenta cada caso

Considerando la medición

Agregando medición al cálculo de van Tonder
(Mi tesis de licenciatura, UNR, 2007)

- El cálculo original no tiene operador de medición (basado en un resultado que muestra que siempre se puede retrasar la medición hasta el último paso)

Considerando la medición

Agregando medición al cálculo de van Tonder
(Mi tesis de licenciatura, UNR, 2007)

- El cálculo original no tiene operador de medición (basado en un resultado que muestra que siempre se puede retrasar la medición hasta el último paso)
- Mi trabajo:
  - Detallar mejor la sintaxis de qubits y compuertas
  - Agregar conjuntos de proyectores, con reglas de reescritura probabilistas
  - Prueba de confluencia (*QPL, 2008*, con Arrighi, Gadella y Grattage)

El cálculo de Selinger y Valiron

Principal idea del cálculo de Selinger y Valiron

Registro cuántico (datos cuánticos) y un flujo de control clásico

El cálculo de Selinger y Valiron

Principal idea del cálculo de Selinger y Valiron
Registro cuántico (datos cuánticos) y un flujo de control clásico

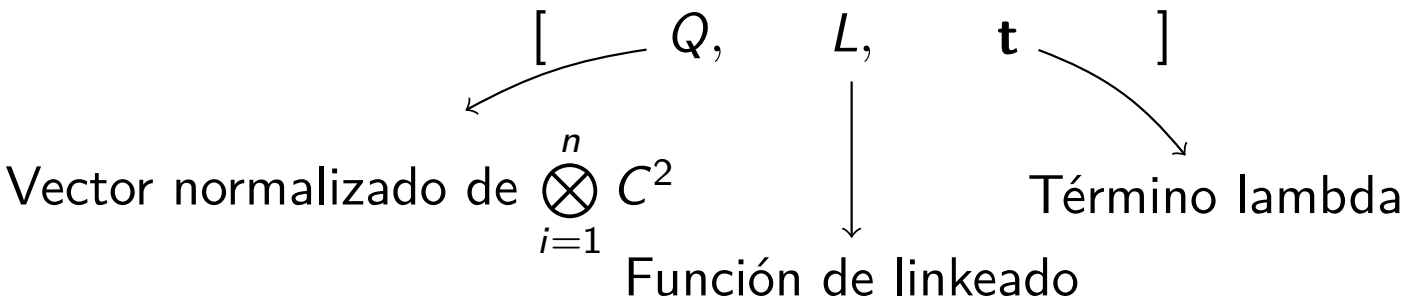
Sintaxis:

$$\begin{aligned} \mathbf{t}, \mathbf{r} \quad ::= \quad & x \mid \lambda x.\mathbf{t} \mid \mathbf{t}\ \mathbf{r} \\ & \mathit{new} \mid \mathit{meas} \mid U \\ & \text{it } \mathbf{t} \text{ then } \mathbf{r} \text{ else } \mathbf{s} \mid 0 \mid 1 \\ & \star \mid (\mathbf{t}_1, \mathbf{t}_2) \mid \text{let } (\mathbf{t}_1, \mathbf{t}_2) = \mathbf{r} \text{ in } \mathbf{s} \end{aligned}$$

- *new*: Mapea un bit clásico a un bit cuántico
- *meas*: Mapea un bit cuántico en uno clásico
- U: Constantes para compuertas unitarias

Programas

Program state

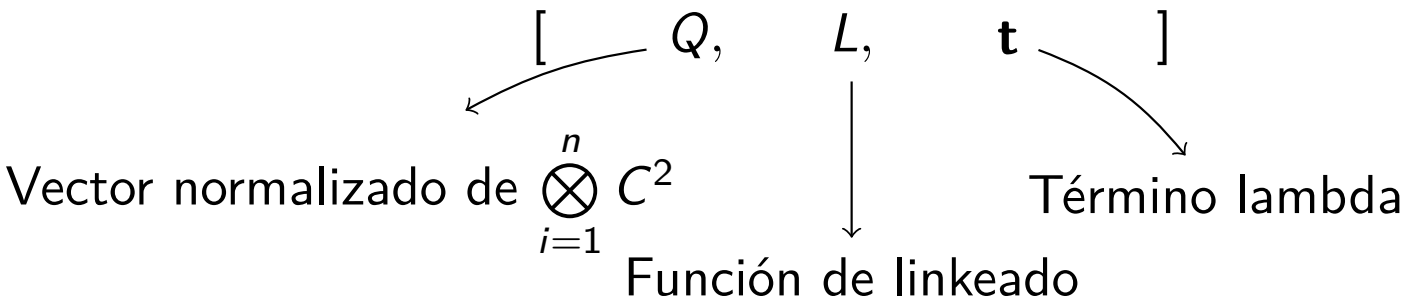


- La función de linkeado linkea variables libres en qubits de Q

  Ejemplo $[\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \{x \mapsto 2\}, \lambda y.x]$

Programas

Program state



- La función de linkeado linkea variables libres en qubits de Q

  Ejemplo $[\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \{x \mapsto 2\}, \lambda y.x]$
- Notación: $[Q, \mathbf{t}[p_{i_1}/x_1] \cdots [p_{i_n}/x_n]]$ si $L(x_k) = i_k$

  Ejemplo $[\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \lambda y.p_2]$

No-clonado == Lógica Lineal

No-clonado $\Rightarrow$ Dos variables no pueden referenciar al mismo qubit

No-clonado == Lógica Lineal

No-clonado $\Rightarrow$ Dos variables no pueden referenciar al mismo qubit

- Sintácticamente: Condición de linearidad:
  $\lambda x.\mathbf{t}$ es LINEAL si x aparece como máximo una vez en $\mathbf{t}$

No-clonado == Lógica Lineal

No-clonado $\Rightarrow$ Dos variables no pueden referenciar al mismo qubit

- Sintácticamente: Condición de linearidad:
  $\lambda x.\mathbf{t}$ es LINEAL si x aparece como máximo una vez en $\mathbf{t}$
- Un programa es bien formado si respeta la condición de linearidad

Condición impuesta por un sistema de tipos basado en LÓGICA LINEAL (que se escapa de este curso)

Estrategia de reducción

$$\textbf{xor} = \lambda x.\lambda y.\text{if } x \text{ then } (\text{if } y \text{ then } 0 \text{ else } 1) \text{ else } (\text{if } y \text{ then } 1 \text{ else } 0)$$

Estrategia de reducción

$$\textbf{xor} = \lambda x.\lambda y.\text{if } x \text{ then (if } y \text{ then 0 else 1)}$$
$$\text{else (if } y \text{ then 1 else 0)}$$

$$[|0\rangle, (\lambda x.\textbf{xor } x\ x)(meas(H\ p_1))]$$

Estrategia de reducción

$$\textbf{xor} = \lambda x.\lambda y.\text{if } x \text{ then (if } y \text{ then 0 else 1)}$$
$$\text{else (if } y \text{ then 1 else 0)}$$

$$[|0\rangle, (\lambda x.\textbf{xor}\ x\ x)(meas(H\ p_1))]$$

En el pizarrón

- Call-by-value
- Call-by-name
- Mixed

Reglas de reducción

$$[Q, (\lambda x.\mathbf{t})\mathbf{v}] \to_1 [Q, \mathbf{t}[\mathbf{v}/x]]$$
$$[Q, \text{if } 0 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \to_1 [Q, \mathbf{r}]$$
$$[Q, \text{if } 1 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \to_1 [Q, \mathbf{t}]$$

$$[Q, U(p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))] \to_1 [UQ, (p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))]$$

$$[Q_n, \mathit{new}\ 0] \to_1 [Q \otimes |0\rangle, p_{n+1}]$$
$$[Q_n, \mathit{new}\ 1] \to_1 [Q \otimes |1\rangle, p_{n+1}]$$

Donde Q_n es un n-qubit

$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \to_{|\alpha|^2} [|\psi_0\rangle, 0]$$
$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \to_{|\beta|^2} [|\psi_1\rangle, 1]$$

Donde ψ_0 tiene $|0\rangle$ en posición i y ψ_1 tiene $|1\rangle$ en i

Reglas de reducción

$$[Q, (\lambda x.\mathbf{t})\mathbf{v}] \rightarrow_1 [Q, \mathbf{t}[\mathbf{v}/x]]$$
$$[Q, \text{if } 0 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \rightarrow_1 [Q, \mathbf{r}]$$
$$[Q, \text{if } 1 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \rightarrow_1 [Q, \mathbf{t}]$$

$$[Q, U(p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))] \rightarrow_1 [UQ, (p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))]$$

$$[Q_n, \mathit{new}\ 0] \rightarrow_1 [Q \otimes |0\rangle, p_{n+1}]$$
$$[Q_n, \mathit{new}\ 1] \rightarrow_1 [Q \otimes |1\rangle, p_{n+1}]$$

Donde Q_n es un n-qubit

$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \rightarrow_{|\alpha|^2} [|\psi_0\rangle, 0]$$
$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \rightarrow_{|\beta|^2} [|\psi_1\rangle, 1]$$

Donde ψ_0 tiene $|0\rangle$ en posición i y ψ_1 tiene $|1\rangle$ en i

Ejemplo (teleportación) en el pizarrón

Reglas de reducción

$$[Q, (\lambda x.\mathbf{t})\mathbf{v}] \rightarrow_1 [Q, \mathbf{t}[\mathbf{v}/x]]$$
$$[Q, \text{if } 0 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \rightarrow_1 [Q, \mathbf{r}]$$
$$[Q, \text{if } 1 \text{ then } \mathbf{t} \text{ else } \mathbf{r}] \rightarrow_1 [Q, \mathbf{t}]$$

$$[Q, U(p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))] \rightarrow_1 [UQ, (p_{j_1}, (p_{j_2}, (\cdots, p_{j_n})))]$$

$$[Q_n, \mathit{new}\ 0] \rightarrow_1 [Q \otimes |0\rangle, p_{n+1}]$$
$$[Q_n, \mathit{new}\ 1] \rightarrow_1 [Q \otimes |1\rangle, p_{n+1}]$$

Donde Q_n es un n-qubit

$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \rightarrow_{|\alpha|^2} [|\psi_0\rangle, 0]$$
$$[\alpha|\psi_0\rangle + \beta|\psi_1\rangle, \mathit{meas}\ p_i] \rightarrow_{|\beta|^2} [|\psi_1\rangle, 1]$$

Donde ψ_0 tiene $|0\rangle$ en posición i y ψ_1 tiene $|1\rangle$ en i

Ejemplo (teleportación) en el pizarrón

Ejercicios Escribir el algoritmo de Deutsch y la codificación superdensa

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá
- **El objetivo**

  Crear un lenguaje de alto nivel que *"estime con exactitud y reduzca los recursos computacionales requeridos para implementar algoritmos cuánticos en una computadora cuántica real".*

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá
- **El objetivo**

  Crear un lenguaje de alto nivel que *"estime con exactitud y reduzca los recursos computacionales requeridos para implementar algoritmos cuánticos en una computadora cuántica real".*
- Pero no existe una "computadora cuántica real" (no escalable)

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá
- **El objetivo**

  Crear un lenguaje de alto nivel que *"estime con exactitud y reduzca los recursos computacionales requeridos para implementar algoritmos cuánticos en una computadora cuántica real".*
- Pero no existe una "computadora cuántica real" (no escalable)
- Quipper se desarrolló teniendo en cuenta lo que HOY es costoso y lo que no (es "fácilmente" modificable)

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá
- **El objetivo**

  Crear un lenguaje de alto nivel que *"estime con exactitud y reduzca los recursos computacionales requeridos para implementar algoritmos cuánticos en una computadora cuántica real".*
- Pero no existe una "computadora cuántica real" (no escalable)
- Quipper se desarrolló teniendo en cuenta lo que HOY es costoso y lo que no (es "fácilmente" modificable)
- Embebido en Haskell (o sea: Quipper = Colección de tipos de datos, combinadores, y librerías de funciones de Haskell)

El caso de Quipper

- Financiado por la Dirección de Inteligencia Nacional de Estados Unidos
- Desarrollado entre 2 universidades de Estados Unidos y una de Canadá
- **El objetivo**

  Crear un lenguaje de alto nivel que *"estime con exactitud y reduzca los recursos computacionales requeridos para implementar algoritmos cuánticos en una computadora cuántica real".*
- Pero no existe una "computadora cuántica real" (no escalable)
- Quipper se desarrolló teniendo en cuenta lo que HOY es costoso y lo que no (es "fácilmente" modificable)
- Embebido en Haskell (o sea: Quipper = Colección de tipos de datos, combinadores, y librerías de funciones de Haskell)
- Fuentes y binarios descargables de `http://www.mathstat.dal.ca/~selinger/quipper`