

# A functional programming language for quantum computation with classical control

Peter Selinger<sup>a</sup> and Benoît Valiron<sup>b</sup>

University of Ottawa, 585 King Edward Avenue,  
K1N 6N5 Ottawa ON, Canada

---

<sup>a</sup>email: [selinger@mathstat.uottawa.ca](mailto:selinger@mathstat.uottawa.ca)

<sup>b</sup>email: [bvali087@uottawa.ca](mailto:bvali087@uottawa.ca)

## Quantum computation

- Quantum bit : orthonormal vector in a 2-dimensional Hilbert space:  $\mathbb{C}^2 \setminus \{0\}/(0, \infty)$ . We choose an orthonormal basis  $\{|0\rangle, |1\rangle\}$ .
- The product of two qubits lives in the tensor product of the two corresponding spaces. A canonical basis for it is  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ , where  $|xy\rangle = |x\rangle \otimes |y\rangle$ .
- The measurement of a qubit  $\alpha|0\rangle + \beta|1\rangle$  returns a bit  $x$  with some probability  $\mu_x$  and collapses the state onto
  - $|0\rangle$  if 0 was measured, and  $\mu_0 = |\alpha|^2$ ;
  - $|1\rangle$  if 1 was measured, and  $\mu_1 = |\beta|^2$ .

## Quantum computation (2)

- Unitary operations on qubits:  $U$  is unitary if  $U^\dagger U = I$ . For example,

$$H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

w.r.t basis  $\{|0\rangle, |1\rangle\}$  and  $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ , respectively.

- No-cloning property: There is no map duplicating all qubit states:  
 $(\alpha|0\rangle + \beta|1\rangle) \mapsto (\alpha|0\rangle + \beta|1\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle)$
- Entanglement:  $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  is a 2-qubit system but cannot be written as  $|p\rangle \otimes |q\rangle$ .

## Functional programming

- Higher order :  $\lambda$ -calculus
- Some terms can be duplicated, others cannot. Information can be:
  - on terms (Abramsky, Wadler) : let the programmer decide.
  - on types : let the compiler decide

For example:

$$x : qbit \triangleright \lambda y. x : bit \Rightarrow qbit$$

$$x : qbit \triangleright \lambda y. \text{new } y : bit \Rightarrow qbit$$

$$\lambda f x. f(\text{meas}(fx)) : (bit \Rightarrow qbit) \Rightarrow bit \Rightarrow qbit$$

## Lambda-calculus: Terms

*Term*  $M, N, P$  ::=  $x$   
|  $(MN)$   
|  $\lambda x.M$   
| *if*  $M$  *then*  $N$  *else*  $P$   
|  $0$  |  $1$   
| *meas*  
| *new*  
|  $U$   
|  $*$   
|  $\langle M, N \rangle$   
| *let*  $\langle x, y \rangle = M$  *in*  $N$

## Encoding qubits in terms

Directly:

$$\lambda x \cdot |1\rangle$$

Problem due to entanglement:

$$\lambda x \cdot (xp)q$$

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

# Program

$$[Q, L, M]$$

where

- $Q$  is a vector in a suitable Hilbert space (Note: if  $Q = 0$ , one writes  $Q = |\rangle$ ),
- $L$  is a linking function,
- $M$  is a lambda-term

We write  $[[1\rangle, p_0]$  in place of  $[[1\rangle, \{x \mapsto 0\}, x]$ .

## Importance of the reduction strategy

Consider

**plus** =  $\lambda xy. \text{if } x \text{ then}(\text{if } y \text{ then } 0 \text{ else } 1) \text{ else}(\text{if } y \text{ then } 1 \text{ else } 0)$

and the program

$[| \rangle, (\lambda x. \mathbf{plus} \ x \ x)(\text{meas}(H(\text{new } 0)))]$

In call-by-value:

$\longrightarrow_{CBV} [|0\rangle, (\lambda x. \mathbf{plus} \ x \ x)(\text{meas}(H \ p_0))]$   
 $\longrightarrow_{CBV} [\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), (\lambda x. \mathbf{plus} \ x \ x)(\text{meas} \ p_0)]$   
 $\longrightarrow_{CBV} [|0\rangle, 0] \quad \text{or} \quad [|1\rangle, 0] \quad (\text{with prob. } 0.5 \text{ each})$

## Importance of the reduction strategy (2)

In call-by-name:

$$[ |\rangle, (\lambda x. \mathbf{plus} \ x \ x)(\mathit{meas}(H(\mathit{new} \ 0)))]$$

reduces to

$$[ |\rangle, \mathbf{plus} (\mathit{meas}(H(\mathit{new} \ 0))) (\mathit{meas}(H(\mathit{new} \ 0)))],$$

$$\longrightarrow_{CBN} \left\{ \begin{array}{ll} [|00\rangle, 0] & \text{with prob. 0.25} \\ [|01\rangle, 1] & \text{with prob. 0.25} \\ [|10\rangle, 1] & \text{with prob. 0.25} \\ [|11\rangle, 0] & \text{with prob. 0.25} \end{array} \right.$$

## Definition: Reduction Strategy

$$\begin{array}{c}
 \overline{[Q, (\lambda x.M)V] \longrightarrow_1 [Q, M[V/x]]} \quad (\beta) \\
 \frac{[Q, N] \longrightarrow_p [Q', N']}{\overline{[Q, MN] \longrightarrow_p [Q', MN']}} \quad (cong_1) \\
 \frac{[Q, M] \longrightarrow_p [Q', M']}{\overline{[Q, MV] \longrightarrow_p [Q', M'V]}} \quad (cong_2) \\
 \\
 \overline{[Q, \text{if } 0 \text{ then } M \text{ else } N)] \longrightarrow_1 [Q, N]} \quad (if_0) \\
 \overline{[Q, \text{if } 1 \text{ then } M \text{ else } N)] \longrightarrow_1 [Q, M]} \quad (if_1) \\
 \frac{[Q, P] \longrightarrow_p [Q', P']}{\overline{[Q, \text{if } P \text{ then } M \text{ else } N)] \longrightarrow_p [Q', \text{if } P' \text{ then } M \text{ else } N]}} \quad (\xi_{if})
 \end{array}$$

## Definition: Reduction Strategy

$$\begin{array}{c}
 \overline{[Q, \text{let } \langle x_1, x_2 \rangle = \langle V_1, V_2 \rangle \text{ in } N] \longrightarrow_1 [Q, N[V_1/x_1, V_2/x_2]]} \\
 \overline{[Q, M_1] \longrightarrow_p [Q', M'_1]} \\
 \overline{[Q, \langle M_1, M_2 \rangle] \longrightarrow_p [Q', \langle M'_1, M_2 \rangle]} \\
 \overline{[Q, M_2] \longrightarrow_p [Q', M'_2]} \\
 \overline{[Q, \langle V_1, M_2 \rangle] \longrightarrow_p [Q', \langle V_1, M'_2 \rangle]} \\
 \overline{[\alpha|Q_0\rangle + \beta|Q_1\rangle, \text{meas } p_i] \longrightarrow_{\mu_0} [|Q_0\rangle, 0]} \quad (\text{meas}) \\
 \overline{[\alpha|Q_0\rangle + \beta|Q_1\rangle, \text{meas } p_i] \longrightarrow_{\mu_1} [|Q_1\rangle, 1]} \quad (\text{meas}) \\
 \overline{[Q, \text{new } 0] \longrightarrow_1 [Q \otimes |0\rangle, p_n]} \quad (\text{new}_0) \\
 \overline{[Q, \text{new } 1] \longrightarrow_1 [Q \otimes |1\rangle, p_n]} \quad (\text{new}_1) \\
 \overline{[Q, U p_j] \longrightarrow_1 [Q', p_j]} \quad (U)
 \end{array}$$

# Types

We need a notion of linearity: We use affine intuitionistic linear logic as a framework.

2 constant types: *bit* and *qbit*.

$$qType\ A, B ::= \alpha \mid X \mid !A \mid (A \multimap B) \mid (A \otimes B)$$

Note: there is no bijection between  $\langle \pi_1 M, \pi_2 M \rangle$  and  $M$ .

# Subtyping

In a term  $M[x : A]$  one can substitute  $x$  by  $y : !A$ .

Notion of subtyping:

$$\frac{}{\alpha <: \alpha} \text{ (cons)} \quad \frac{A <: B}{!A <: B} \text{ (D)} \quad \frac{}{X <: X} \text{ (var)}$$

$$\frac{!A <: B}{!A <: !B} \text{ (!)} \quad \frac{A <: A' \quad B <: B'}{A' \multimap B <: A \multimap B'} \text{ (}\multimap\text{)}$$

$$\frac{A <: A' \quad B <: B'}{A \otimes B <: A' \otimes B'} \text{ (}\otimes\text{)}$$

Note: we write  $!^n A$  for  $!!! \dots !!! A$

## Types for constant terms

We know what type the constant terms should have. Let us fix a type assignment  $c \mapsto A_c$ , from the set of constant terms to  $qType$ :

$$A_0 = !bit$$

$$A_1 = !bit$$

$$A_{new} = !(bit \multimap qbit)$$

$$A_U = !(qbit \multimap qbit) \quad A_{meas} = !(qbit \multimap !bit)$$

## Typing rules

For  $c$  a constant term:

$$\frac{A <: B}{\Delta, x : A \triangleright x : B} \text{ (ax}_1\text{)} \quad \frac{A_c <: B}{\Delta \triangleright c : B} \text{ (ax}_2\text{)}$$

For the *if* term:

$$\frac{\Gamma_1, !\Delta \triangleright P : \textit{bit} \quad \Gamma_2, !\Delta \triangleright M : A \quad \Gamma_2, !\Delta \triangleright N : A}{\Gamma_1, \Gamma_2, !\Delta \triangleright \textit{if } P \textit{ then } M \textit{ else } N : A} \text{ (if)}$$

The application:

$$\frac{\Gamma_1, !\Delta \triangleright M : A \multimap B \quad \Gamma_2, !\Delta \triangleright N : A}{\Gamma_1, \Gamma_2, !\Delta \triangleright MN : B} \text{ (app)}$$

The lambda, where  $x \notin |\Delta|$ :

$$\frac{x : A, \Delta \triangleright M : B}{\Delta \triangleright \lambda x. M : A \multimap B} \text{ (\lambda}_1\text{)}$$

If  $FV(M) \cap |\Gamma| = \emptyset$ :

$$\frac{\Gamma, !\Delta, x : A \triangleright M : B}{\Gamma, !\Delta \triangleright \lambda x. M : !^{n+1}(A \multimap B)} \text{ (\lambda}_2\text{)}$$

## Typing Rules (2)

The pair and the *let* term:

$$\frac{! \Delta, \Gamma_1 \triangleright M : !^n (A_1 \otimes A_2) \quad ! \Delta, \Gamma_2, x_1 : !^n A_1, x_2 : !^n A_2 \triangleright N : A}{! \Delta, \Gamma_1, \Gamma_2 \triangleright \text{let } \langle x_1, x_2 \rangle = M \text{ in } N : A} \quad (\otimes.E)$$

$$\frac{! \Delta, \Gamma_1 \triangleright M_1 : !^n A_1 \quad ! \Delta, \Gamma_2 \triangleright M_2 : !^n A_2}{! \Delta, \Gamma_1, \Gamma_2 \triangleright \langle M_1, M_2 \rangle : !^n (A_1 \otimes A_2)} \quad (\otimes.I)$$

## Some results.

### Lemma 1

$$\frac{\Gamma \triangleleft: \Delta \quad \Delta \triangleright N : A \quad A \triangleleft: B}{\Gamma \triangleright N : B}$$

**Lemma 2** *If  $V$  is a value such that  $\Delta \triangleright V : !A$ ,*

$$\forall x \in FV(M) \quad \exists U \quad \Delta_f(x) = !U$$

**Lemma 3** *If  $V$  is a value,*

$$\frac{! \Delta, \Gamma_1, x : A \triangleright M : B \quad ! \Delta, \Gamma_2 \triangleright V : A}{\Gamma_1, \Gamma_2, ! \Delta \triangleright M[V/x] : B}$$

## Substitution Lemma

**Lemma 4 (Substitution)** *If  $V$  is a value,*

$$\frac{\Gamma_1, !\Delta, x : A \triangleright M : B \quad \Gamma_2, !\Delta \triangleright V : !^n A}{\Gamma_1, \Gamma_2, !\Delta \triangleright M[V/x] : B}$$

## Safety properties

**Theorem 1 (Subject Reduction)** *If  $[Q, L, M] : A$  reduces to  $[Q', L', M']$  then  $[Q', L', M'] : A$ .*

**Theorem 2 (Progress)** *Let  $[Q, L, M]$  be a valid program, then either it is a value or it reduces.*



## Quantum vs. simply-typed lambda-calculus

$$\pi = \frac{\frac{\frac{\overline{x:!(A \multimap A) \triangleright x:A \multimap A} \quad \frac{\overline{y:A \triangleright y:A}}{\overline{x:!(A \multimap A) \triangleright xy : A}}}{\overline{x:!(A \multimap A), y:A \triangleright x(xy) : A}}}{\overline{x:!(A \multimap A) \triangleright \lambda y.x(xy):!(A \multimap A)}}}{\triangleright \lambda xy.x(xy):!(A \multimap A) \multimap !(A \multimap A)}$$

$$\dagger \pi = \frac{\frac{\frac{\overline{x:(A \Rightarrow A) \blacktriangleright x:A \Rightarrow A} \quad \frac{\overline{y:A \blacktriangleright y:A}}{\overline{x:(A \Rightarrow A) \blacktriangleright xy : A}}}{\overline{x:(A \Rightarrow A), y:A \blacktriangleright x(xy) : A}}}{\overline{x:(A \Rightarrow A) \blacktriangleright \lambda y.x(xy):(A \Rightarrow A)}}}{\blacktriangleright \lambda xy.x(xy):(A \Rightarrow A) \Rightarrow (A \Rightarrow A)}$$

$\dagger \pi$  is called the *skeleton* of  $\pi$ .

## Linear Decorations

Given  $A$  an intuitionistic type,  $U$  a quantum type, we define  $A \multimap U$ , decoration of  $A$  along  $U$ , by:

$$\begin{aligned} A \multimap !U &= !(A \multimap U) \\ (A \Rightarrow B) \multimap (U \multimap V) &= (A \multimap U) \multimap (B \multimap V) \\ (A \times B) \multimap (U \otimes V) &= (A \multimap U) \otimes (B \multimap V) \\ \text{in all other case, } A \multimap U &= A^* \end{aligned}$$

where

$$\begin{aligned} (A \Rightarrow B)^* &= A^* \multimap B^* \\ (A \times B)^* &= A^* \otimes B^* \\ X^* &= X \\ \alpha^* &= \alpha \end{aligned}$$

## Linear Decorations

What we want:

**Property 1 (desired)** *Given  $M$  quantum typeable, any intuitionistic derivation is the skeleton of a quantum one.*

Unfortunately,

$$\frac{x : \mathit{qbit} \blacktriangleright \mathit{meas} : \mathit{qbit} \Rightarrow \mathit{bit} \quad x : \mathit{qbit} \blacktriangleright x : \mathit{qbit}}{\frac{x : \mathit{qbit} \blacktriangleright \mathit{meas} x : \mathit{bit}}{\blacktriangleright \lambda x. \mathit{meas} x : \mathit{qbit} \Rightarrow \mathit{bit}}}$$

is valid but not the skeleton of anything.

The problem is

$$\frac{\boxed{x : \mathit{qbit}} \blacktriangleright \mathit{meas} : \mathit{qbit} \Rightarrow \mathit{bit} \quad x : \mathit{qbit} \blacktriangleright x : \mathit{qbit}}{x : \mathit{qbit} \blacktriangleright \mathit{meas} x : \mathit{bit}} \quad \blacktriangleright \lambda x. \mathit{meas} x : \mathit{qbit} \Rightarrow \mathit{bit}$$

## Normal form

**Definition.** The normal form of a typing judgement is

$$\mathcal{N}(\Delta \triangleright M : A) = (\Delta|_{FV(M)} \triangleright M : A)$$

$$\mathcal{N}(\Delta \blacktriangleright M : A) = (\Delta|_{FV(M)} \blacktriangleright M : A)$$

And now,

**Theorem 3** *If  $M$  is quantum typeable, any **normal** intuitionistic derivation  $\pi'$  is the skeleton of a quantum one.*

## Fundamental Lemma

**Lemma 5** *If  $\pi$  is a normal intuitionistic type derivation and if  $\rho$  is any quantum type derivation, then  $\pi' := (\pi \multimap \rho)$  is a normal quantum type derivation.*

## Result: Inference Algorithm

We can perform type inference in the quantum lambda-calculus in three steps:

1. Find an intuitionistic type derivation  $\pi$ , if any.
2. Reduce it to its normal form  $\mathcal{N}\pi$ .
3. Find a decoration of  $\mathcal{N}\pi$  which is a valid quantum type derivation, if any.

Moreover, using the fact that  $!!A \doteq !A$ , we can only consider decorations without repeated exponentials.

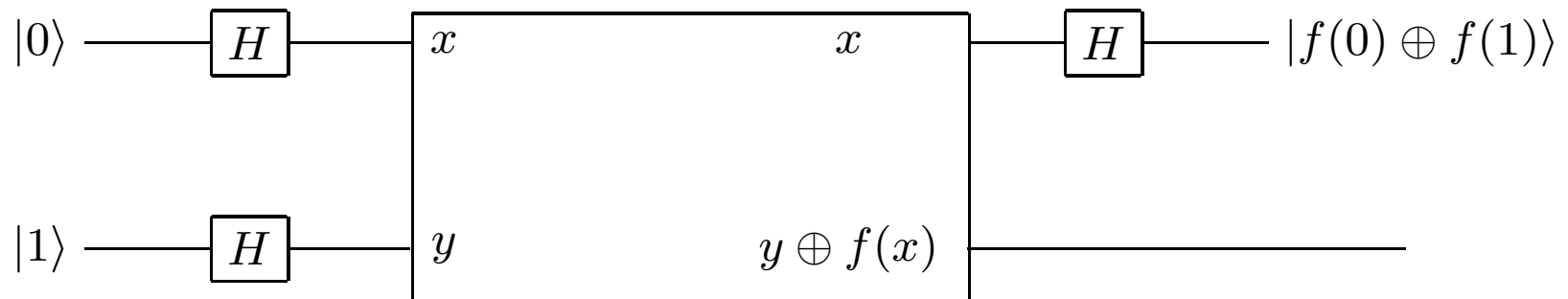
## Conclusion

- A quantum  $\lambda$ -calculus with classical features
- Verify the safety properties:
  - Subject Reduction
  - Progress Lemma
- Inference algorithm

## Future Work

- Recursion, additive types
- Call-by-name case
- Relation with affine intuitionistic linear logic
- Denotational semantics

## Example: the Deutsch Algorithm



let  $\mathbf{F} U_f =$

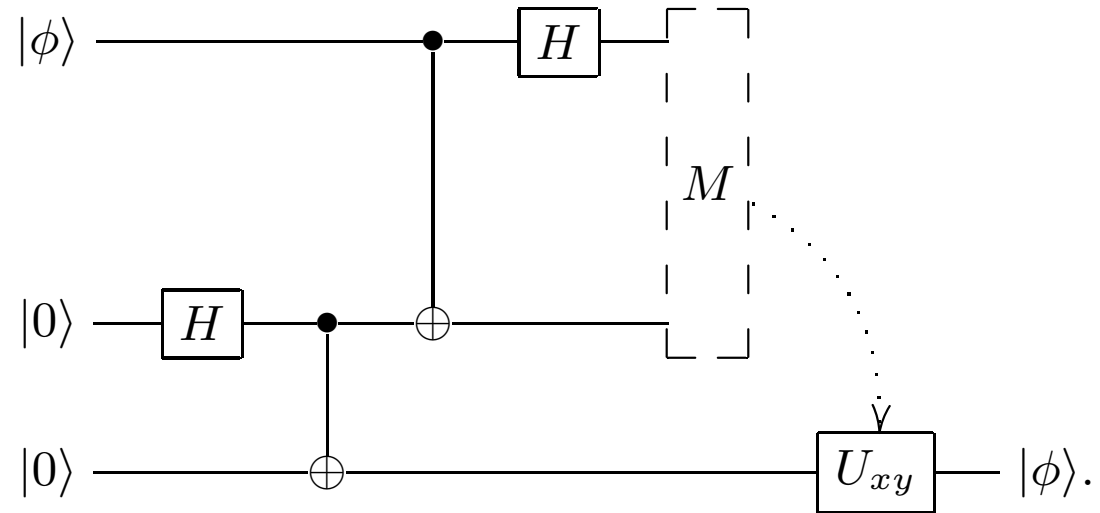
let  $\mathbf{tens} f g \langle x, y \rangle = \langle fx, gy \rangle$

in let  $\langle x, y \rangle =$

$(\mathbf{tens} H (\lambda x.x))(U_f \langle H(\mathit{new} 0), H(\mathit{new} 1) \rangle)$

in meas  $x$

## The teleportation procedure



$$\begin{aligned}
 U_{00} &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & U_{01} &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
 U_{10} &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} & U_{11} &= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.
 \end{aligned}$$

## The teleportation procedure

**EPR** ::  $!(\top \multimap (\text{qbit} \otimes \text{qbit}))$

**EPR** =  $\lambda x. \text{CNOT} \langle H(\text{new } 0), \text{new } 0 \rangle$

**BellMeasure** ::  $!(\text{qbit} \multimap (\text{qbit} \multimap \text{bit} \otimes \text{bit}))$

**BellMeasure** =  $\lambda q_2. \lambda q_1. (\text{let } \langle x, y \rangle = \text{CNOT} \langle q_1, q_2 \rangle$   
 $\text{in } \langle \text{meas}(Hx), \text{meas } y \rangle)$

**U** ::  $!(\text{qbit} \multimap (\text{bit} \otimes \text{bit} \multimap \text{qbit}))$

**U** =  $\lambda q. \lambda \langle x, y \rangle. \text{if } x \text{ then } (\text{if } y \text{ then } U_{11}q \text{ else } U_{10}q)$   
 $\text{else } (\text{if } y \text{ then } U_{01}q \text{ else } U_{00}q),$

## The teleportation procedure

We construct two non-duplicable functions  $f : \text{qbit} \multimap \text{bit} \otimes \text{bit}$  and  $g : \text{bit} \otimes \text{bit} \multimap \text{qbit}$ , such that  $g \circ f(x) = x$  for an arbitrary qubit  $x$ .

*let*  $\langle x, y \rangle = \mathbf{EPR} *$

*in let*  $f = \mathbf{BellMeasure} \ x$

*in let*  $g = \mathbf{U} \ y.$

*in*  $\langle f, g \rangle.$