Categorical Semantics for Proto-Quipper Language and Dynamic Lifting

Abu Dhabi, April 20th 2024

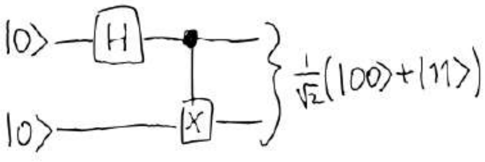
Dongho Lee

Table of Contents

Table of Contents

Quantum Circuit Model

Quantum circuit model



Quantum states: defined in Hilbert space

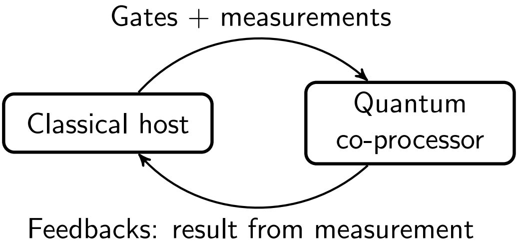
- qubit: 2-dimensional Hilbert space
- multiple qubits: tensor product of Hilbert spaces
- pure state: normalized vector in Hilbert space
- computational basis: classical state

Quantum operators: transformation between quantum states

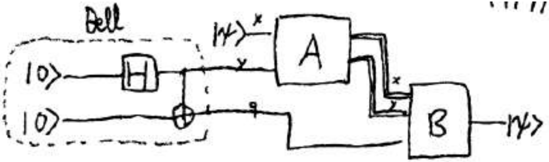
- Unitary maps

QRAM Model

QRAM model of classical computer and quantum co-processor:



Quantum processes use measurement and classical control flow



Quantum computation can be statistical set of quantum computation

Characteristics of QRAM model

- Quantum operators: unitary maps, initialization, measurement
- Mixed state:
  - probability distribution over pure states

Example (Measurement of a qubit over computational basis)

$$\mathrm{meas}_0(v) = v^*(|0\rangle\langle 0|)v = |\alpha|^2 = p_0$$
$$\mathrm{meas}_1(v) = v^*(|1\rangle\langle 1|)v = |\beta|^2 = p_1$$

where $v = \alpha|0\rangle + \beta|1\rangle$.

- Quantum computation can give advantage in computation
  - Caveat: no-cloning theorem

Quantum programming language based on QRAM model-1

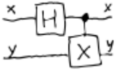
- Quantum λ-calculus
  - higher-order functional language
  - quantum types and quantum operators
  - linear type system with exponential constructor ! for classical types
  - allows the classical control flow introduced by measurements

Example (Duplicable and non-duplicable terms)

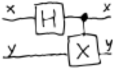
(cointoss)	$\vdash$	$\text{meas}(H(\text{init tt}))$	$:!(\top \multimap \text{bool})$
(entangle)	$\vdash$	$\lambda x.\text{CX}\langle x, \text{init tt}\rangle$	$:!(\textbf{qubit} \multimap \textbf{qubit} \otimes \textbf{qubit})$
(entangle')	$v : \textbf{qubit} \vdash$	$\text{CX}\langle v, \text{init tt}\rangle$	$: \textbf{qubit} \otimes \textbf{qubit}$

Quantum programming language based on QRAM model-2

- Quantum circuit description languages
  - two-layers of compilation
  - quantum circuits are first-class objects with type $\mathbf{Circ}(A, B)$
  - circuit-level operators
    - box operator (box $:!(A \multimap B) \multimap !\mathbf{Circ}(A, B)$) transforms a function into circuit constant

$$\text{box}(\lambda x.\lambda y.\text{CX}(H(x), y)) = (x \otimes y,$$  $$, x \otimes y)$$

    - unbox operator (unbox : $\mathbf{Circ}(A, B) \multimap (A \multimap B)$) sends the circuit object to the quantum co-processor

$$\text{unbox}(x \otimes y,$$  $$, x \otimes y)(a \otimes b) = a \otimes b$$

  - Examples: Quipper and QWire

Bringing quantum computation into logic

- Problem of realizability of programs

$$\lambda x.\langle x, x\rangle : \text{violates no-cloning theorem}$$
$$\lambda f.\lambda x.f(f(x)) : \text{valid}$$

when applied to qubit x and quantum circuit f.

- Linear logic: resource sensitive logic
  - quantum state is linear resource
  - quantum circuit is non-linear resource

- Curry-Howard correspondence between proof and computation

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (\otimes_R) \qquad \frac{\Gamma \vdash x : A \qquad \Delta \vdash y : B}{\Gamma, \Delta \vdash \langle x, y\rangle : A \otimes B}\ (\otimes_R)$$

Multiplicative exponential of linear logic

- Multiplicative/exponential fragment of Intuitionistic Linear Logic

$$A ::= p \mid I \mid A \otimes A \mid A \multimap A \mid !A$$

- Example: inference rules

$$\frac{!\Gamma \vdash A}{!\Gamma \vdash !A}\ (\mathrm{Pr}) \quad \frac{\Gamma, A \vdash B}{\Gamma !A \vdash B}\ (\mathrm{De}) \quad \frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B}\ (\mathrm{Con}) \quad \frac{\Gamma \vdash B}{\Gamma, !A \vdash B}\ (\mathrm{We})$$

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ (\otimes_R) \quad \frac{\Gamma A, B \vdash C}{\Gamma, A \otimes B \vdash C}\ (\otimes_L)$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ (\multimap_R) \quad \frac{\Gamma \vdash A \qquad \Delta, B \vdash C}{\Gamma, \Delta, A \multimap B \vdash C}\ (\multimap_L)$$

Example of proof in linear logic

- $p \vdash p \otimes p$ is not derivable
- $!(p \multimap p) \vdash p \multimap p$ is derivable

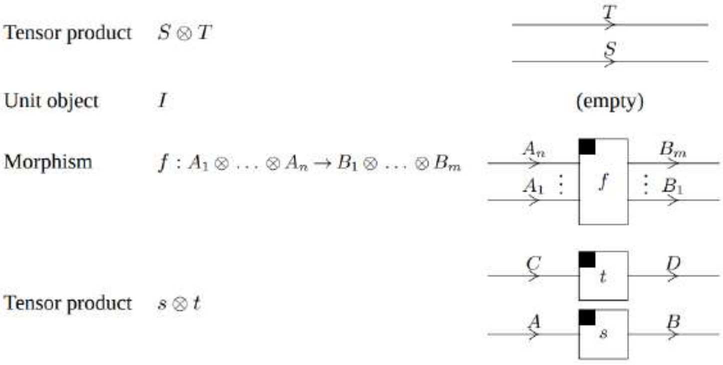
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p \vdash p \qquad p \vdash p}{p,\ p \multimap p \vdash p}\ (\multimap_L) \qquad p \vdash p}{p,\ p \multimap p,\ p \multimap p \vdash p}\ (\multimap_L)}{p \multimap p,\ p \multimap p \vdash p \multimap p}\ (\multimap_R)}{!(p \multimap p),\ !(p \multimap p) \vdash p \multimap p}\ (\mathit{Dereliction})}{!(p \multimap p) \vdash p \multimap p}\ (\mathit{Contraction})$$

Semantics of quantum programming languages

- Operational semantics
  - interprets program as a sequence of configurations
  - gives intuitive formalization of computation
  - hard to analyze the behaviour of programs
- Denotational semantics
  - interpretes program in compositional manner
  - comparison of programs
- Categorical semantics of programming language
  - object $[\![A]\!] \iff$ type A
  - arrow $[\![\Gamma]\!] \xrightarrow{[\![m]\!]} [\![A]\!] \iff \Gamma \vdash m : A$
- Properties of denotational semantics
  - observational equivalence of terms
  - Soundness: equality of terms implies equality of denotation
  - Adequacy: equality of denotation implies equality of terms
  - Fully abstraction: soundness $\wedge$ adequacy

Symmetric monoidal category and diagrams

Graphical language (quantum circuit)	(Symmetric) monoidal category
diagrams consist of gates and wires vertical and horizontal composition	monoidal product monoidal unit

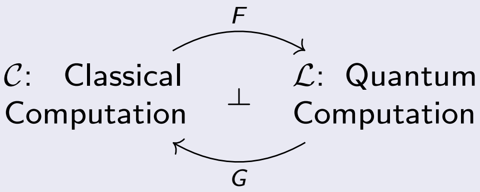


Benton's linear-non-linear category

Definition (Benton's linear-non-linear category)

A linear/non-linear category consists of

- a symmetric monoidal closed category $(\mathcal{L}, \otimes, I, \multimap)$;
- a cartesian closed category $(\mathcal{C}, \times, 1, \rightarrow)$;
- symmetric monoidal adjunction between symmetric monoidal functors $(F, m) : \mathcal{C} \rightarrow \mathcal{L}$ and $(G, n) : \mathcal{L} \rightarrow \mathcal{C}$.



Lemma (Benton)

Every linear-non-linear category is a model of intuitionistic linear logic.

Proto-Quipper-M by Rios and Selinger

- Two levels of execution: state depends on parameter
  - parameters are known at circuit generation time (e.g. bool $= \{0, 1\}$)
  - states are known at circuit execution time (e.g. **qubit**)
- Construction of the model
  - monoidal closed category $\overline{M}$ of quantum circuits
  - coproduct completion $\overline{\overline{M}}$ of $\overline{M}$: $(\# : \overline{\overline{M}} \to \mathbf{Set})$

$$p(\text{bool}) = (\{0, 1\}, (I, I)), \ \mathbf{qubit} = (\{0\}, (\mathbf{qubit}))$$

  - Benton's linear-non-linear category: $(! = p \circ \flat)$

$$\mathbf{Set} \underset{\flat}{\overset{p}{\underset{\bot}{\rightleftarrows}}} \overline{\overline{\mathcal{M}}}$$

- Box and unbox: $\flat(T \multimap U) \cong M(T, U)$

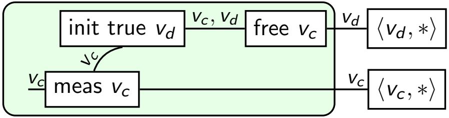
Table of Contents

Semantics of measurement and classical control

Dynamic lifting: information from quantum co-processor to classical host

$$\text{exp} \quad ::= \quad \textbf{let } \langle b, v_c \rangle = \text{meas}(v_c) \textbf{ in} \quad \textbf{if } b \textbf{ then } \langle \text{init}(\text{tt}), \text{free}(v_c) \rangle \textbf{ else } \langle v_c, * \rangle \tag{1}$$

Quantum circuit construction is dependent on the measurement.



Question: How to formalize dynamic lifting in circuit description language?

Make circuits not only lists but trees (*quantum channels*)

Algebraic structure of quantum channel

QCAlg: abstract structure of quantum channels

$$Q \quad ::= \quad \epsilon(W) \ \mid \ U(\vec{W})\ Q \ \mid \ \text{init}\ b\ w\ Q \ \mid \ \text{meas}\ w\ Q_1\ Q_2 \ \mid \ \text{free}\ w\ Q$$

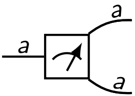
Example (Valid and invalid quantum channels)

- $\epsilon(\{x, y\})$
- $H(x)\ \epsilon(\{x, y\})$
- $\text{init tt}\ x\ \epsilon(\emptyset)$
- $\text{meas}\ x\ \epsilon(\{x\})\ \epsilon(\{x, y\})$
- $\text{free}\ x\ \epsilon(\{x\})$

Quantum channel constant

Quantum channel constant: (p, Q, m)

$(a,$ 

$, m_1)$ $(a,$ 

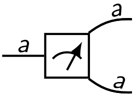
$, m_2)$

Quantum program with dynamic lift may reduce to different values

Type $\mathrm{QChan}(-, -)$ for quantum channel constants.
Each term of branching term has the same type.

$(a,$ 

$, a) : \mathrm{QChan}(\mathbf{qubit}, \mathbf{qubit})$

$(a,$ 

$, \begin{bmatrix} \langle \mathrm{tt}, a \rangle \\ \langle \mathrm{ff}, a \rangle \end{bmatrix}) : \mathrm{QChan}(\mathbf{qubit}, \mathrm{bool} \otimes \mathbf{qubit})$

Proto-Quipper-L - language and type system

Language: non-branching-and branching-term

$$\begin{aligned} \text{Term}(M) &::= \lambda\text{-calculus} \mid (p, Q, m) \mid \text{box}_P \mid \text{unbox} \\ \text{Branching term}(m) &::= M \mid [m_a, m_b] \end{aligned}$$

Linear/non-linear type system ensures quantum variables are used exactly once in each branch of control flow

Type rules ensure that all terms of a branching term share the same type.

Type rules for box and unbox operators:

$$\frac{}{!\Delta \vdash \text{box}_P : !(P \multimap A) \multimap !\text{QChan}(P, A)}(\text{box}) \qquad \frac{}{!\Delta \vdash \text{unbox} : \text{QChan}(P, A) \multimap (P \multimap A)}(\text{unbox})$$

Operational semantics

Configuration is represented by a pair (Q, m) consisting of a quantum channel object Q and a branching term m.

We use a graphical representation of configuration where a green box represents a quantum channel whose leaves are linked to square-boxed terms. The edges represent bundles of wires, which can contain multiple wires and can be empty.

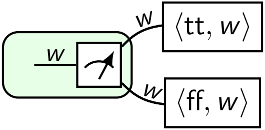
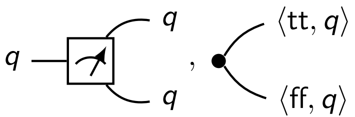


Figure: Graphical representation of a configuration with measurement

Reduction takes place in each branch of branching term.

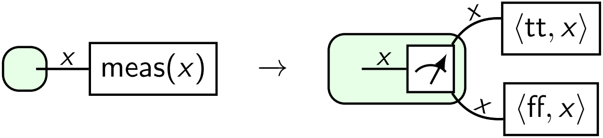
Example - semantics of measurement

A constant meas of type qubit $\multimap$ bool $\otimes$ qubit is defined as:

meas ::= unbox (q, )

Formally by the operational semantics:

$$\frac{\textbf{shape}(\epsilon(\{x\})) = \textbf{shape}(\text{meas}(x))}{}$$



Behavior of measurement:

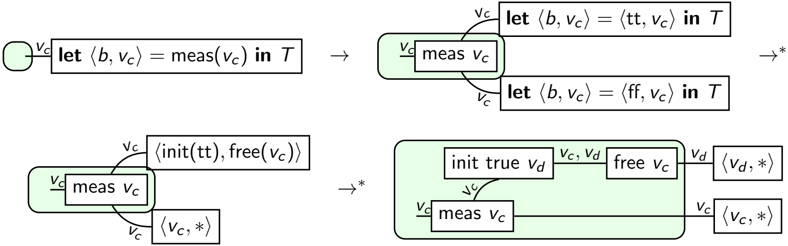
- Classical computation: creates states for each outcome
- Quantum channel: add a measurement gate to the buffer

Example - non-trivial circuit construction

The term in Eq. (1) measures the qubit v_c and construct circuits depending on the measurement.

$$\begin{array}{rcl} \text{exp} & ::= & \textbf{let } \langle b, v_c \rangle = \text{meas}(v_c) \textbf{ in } T \\ T & ::= & \textbf{if } b \textbf{ then } \langle \text{init}(\text{tt}), \text{free}(v_c) \rangle \textbf{ else } \langle v_c, * \rangle \end{array}$$

Despite simple structure, the term does not correspond to a circuit because of the classical control.



Outline of categorical semantics

1. Concrete category of diagrams
2. Proto-Quipper-M by Francisco Rios and Peter Selinger
3. Moggi's categorical model of side-effect with branching monad
   - Monad (F, η, μ) over category $\mathcal{C}$:
     - functor $F : \mathcal{C} \to \mathcal{C}$
     - two natural transformations $\eta : \mathbf{1}_{\mathcal{C}} \to F$ and $\mu : F^2 \to F$
     - monad laws
   - Pure function $p : A \to B$ to function with side-effect $p' : A \to FB$
   - Monad for branching trees

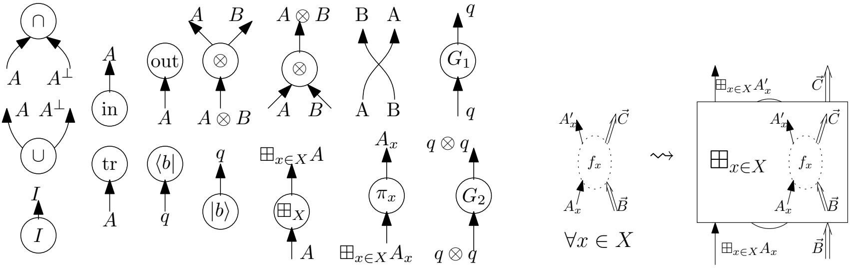
Diagram

Diagram is directed graph, composed of multiple types of nodes and marked edges (node can be a boxed-node of diagrams).

Marks for edge:

$$M \ ::= \ I \ | \ q \ | \ M \otimes M \ | \ \boxplus_{i \in X} M_i \ | \ M^{\perp}$$

Different types of nodes:



(a) Elementary nodes (b) Box node

Equivalence relation of diagrams: example compact closed structure

Category of diagrams ($\overline{M}$):

- object: lists of marks $\vec{A} = [A_1, \ldots, A_n]$, and
- morphism $\vec{A} \to \vec{B}$: equivalence classes of diagrams from $\vec{A}$ to $\vec{B}$

$\overline{M}$ is symmetric monoidal closed:

- The unit: $I = []$,
- $[A_1, \ldots, A_n] \otimes [B_1, \ldots, B_m] = [A_1, \ldots, A_n, B_1, \ldots, B_m]$, and
- $f \otimes g$: juxtaposition of diagrams.
- Internal hom ($\multimap$): application

$$\vec{A} \multimap \vec{B} = [A_1, \ldots, A_n] \multimap [B_1, \ldots, B_m] ::= [A_1^{\perp}, \ldots, A_n^{\perp}, B_1, \ldots, B_m]$$

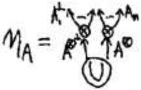
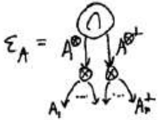
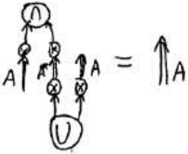
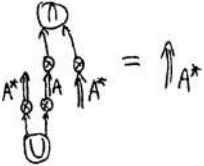
Product ($\times$): $\times_{x \in X} \vec{A_x} = [\boxplus_{x \in X} \vec{A_x}^{\otimes}]$ for any family of objects $(A_x)_{x \in X}$

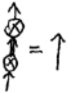
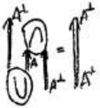
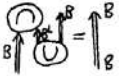
The object $[I \boxplus I]$ corresponds to bit-type.

$\overline{M}$ is a compact closed category

Compact closed category:

- dual of object $\vec{A} = [A_1, \ldots, A_n]$: $\vec{A}^* = [A_1^{\perp}, \ldots, A_n^{\perp}]$
- natural isomorphism $\overline{M}(A, B) \cong A^* \otimes B$:

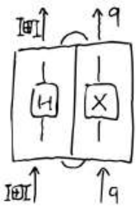
unit  $: I \to \vec{A}^* \otimes A$:	counit  $: \vec{A} \otimes \vec{A}^* \to I$:
	

from  and  

Coproduct completion

Coproduct completion models families of diagrams:

Example (Control flow in parameterized diagrams)



$\cong (\{0,1\},$ ( , ) $)$

$\overline{\overline{M}}$ is symmetric monoidal closed, and features products and co-products.

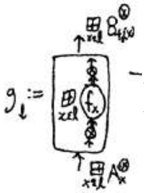
- monoidal unit is $(\{\emptyset\}, (I))$
- $A \otimes B = (X \times Y, (A_x \otimes B_y)_{(x,y)})$
- $A \multimap B = (X \to Y, (C_f)_{f \in X \to Y})$
  (C_f refers to the product $\boxplus_{x \in X}(A_x \multimap B_{f(x)})$ of internal homs in $\overline{M}$)

Monad for Branching Computation

Strong monoidal functor $F : \overline{\overline{M}} \to \overline{\overline{M}}$ of non-deterministic branching effect:

- for an object $A = (X, (A_x))$, $F(A) = (\mathrm{mset}(X), ([\boxplus_{x \in l} A_x^{\otimes}])_{l \in \mathrm{mset}(X)})$,
- for a morphism $f = (f_0, (f_x)) : A \to B$,

$$F(f) = (g_0 : \mathrm{mset}(X) \to \mathrm{mset}(Y),$$



$$)$$

Example (Lifting)

The lifting of the bit $b_s = (\{\emptyset\}, (I \boxplus I))$ to the boolean $b_p = (\{\mathrm{tt}, \mathrm{ff}\}, (I, I))$ is defined as a morphism $\mathrm{lb} : b_s \to F(b_p)$

$$\mathrm{lb} = (\{\emptyset \mapsto [\mathrm{tt}, \mathrm{ff}]\}, (\mathrm{id}_{I \boxplus I}))$$

Interpreting Typed Terms and Configurations

Interpretation of Proto-Quipper-L within the Kleisli category $\overline{\overline{M}}_F$:

- types are mapped to objects;
- typing derivations represent specific morphisms.

The interpretation $[\![A]\!]$ of a type A is built against the categorical structure:

$$[\![I]\!] = (\{\emptyset\}, (I)),\ [\![\mathrm{bool}]\!] = (\{\mathrm{tt}, \mathrm{ff}\}, (I, I))$$

$$[\![\mathbf{qubit}]\!] = (\{\emptyset\}, ([q])),\ [\![A_a \multimap A_b]\!] = [\![A_a]\!] \multimap_{\overline{\overline{M}}_F} [\![A_b]\!]$$

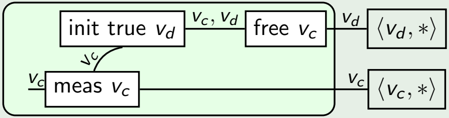
$$[\![A_a \otimes B_b]\!] = [\![A_a]\!] \otimes [\![A_b]\!],\ [\![!A]\!] = ![\![A]\!] = (p \circ b)[\![A]\!]$$

$$[\![\mathrm{QChan}(P, A)]\!] = p(\overline{\overline{M}}_F([\![P]\!], [\![A]\!]))$$

A typed configuration $!\Delta \vdash (Q, m) : A$ is interpreted as the composition of $[\![Q]\!]$ (i.e. we first “compute” Q) followed by the interpretation of m.

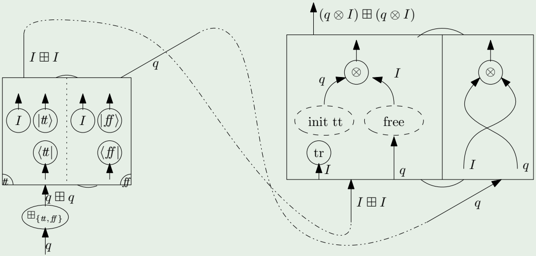
Example (Interpretation of the term in Eq. (1))

The term in Eq.(1)



has for interpretation a morphism

$$(f_0 = \{\{\emptyset\} \mapsto [(\emptyset, \emptyset), (\emptyset, \emptyset)]\}, (f : [q] \mapsto [(q \otimes I) \boxplus (q \otimes I)]))$$



Soundness of the categorical semantics

Denotation of typing derivation is preserved over the reduction.

Theorem (Soundness)

For any configurations (Q_1, m_1) and (Q_2, m_2):

$$\vdash (Q_1, m_1) : A \rightsquigarrow^{*} \vdash (Q_2, m_2) : A$$

(Soundness of operational Semantics)

$$\forall \pi_1 . \exists \pi_2 . \left[\!\left[\frac{\pi_1}{\vdash (Q_1, m_1) : A} \right]\!\right] = \left[\!\left[\frac{\pi_2}{\vdash (Q_2, m_2) : A} \right]\!\right]$$

(Soundness of categorical semantics)

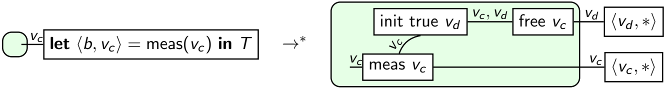
Values of basic types have unique type derivation.

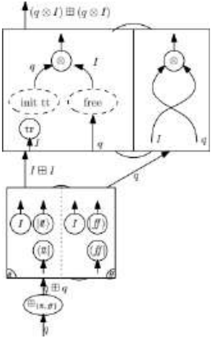
Corollary

All typing derivations of a closed term of basic type have the same interpretation. □

Soundness of the categorical semantics

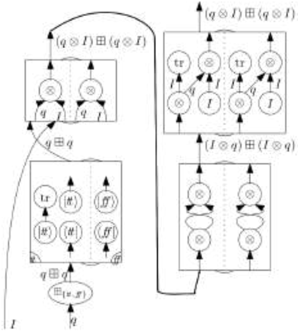
The term in Eq. (1) has the following reduction.





(a) $[\![\vdash (\epsilon(v_c), \text{exp}) : \textbf{qubit} \otimes I]\!]$

=



(b) $[\![\vdash (\text{meas } v_c \ (\text{init } tt \ v_d \ (\text{free } v_c \ (\epsilon(v_d))))(\epsilon(v_c)), [\langle v_d, *\rangle, \langle v_c, *\rangle]) : \textbf{qubit} \otimes I]\!]$

Table of Contents

Proto-Quipper-Dyn by Fu, Kishida, Ross, and Selinger

$\mathcal{V}$-category $\mathcal{A}$ is a linear-non-linear programming language model if:

- $\mathcal{A}$ has coproducts and is symmetric monoidal closed
- $\mathcal{V}$ is cartesian closed and has coproducts
- $\mathcal{V} \underset{\flat}{\overset{p}{\underset{\perp}{\rightleftarrows}}} \mathcal{A}$

$\mathcal{A}$ supports box-unbox operations if:

- There is a fully faithful embedding $\psi : \mathcal{M} \to V(\mathcal{A})$
- For any objects S, U in the image of ψ, $\flat(S \multimap U) \cong \mathcal{A}(S, U)$

Proto-Quipper-Dyn by Fu, Kishida, Ross, and Selinger

$\mathcal{A}$ has dynamic lifting monad $T : \mathcal{A} \to \mathcal{A}$ if:

- T is commutative strong $\mathcal{V}$-monad
- $V(\mathcal{A})_{VT}$ is enriched in convex space
- $$\begin{array}{ccc} \mathcal{M}(S,U) & \xrightarrow{\psi_{S,U}} & V(\mathcal{A})(S,U) \\ {\scriptstyle J_{S,U}}\downarrow & & \downarrow{\scriptstyle \eta} \\ \mathcal{Q}(S,U) & \xrightarrow{\phi_{S,U}} & V(\mathcal{A})_{VT}(S,U) \end{array} \quad \text{and} \quad \begin{array}{ccc} \mathrm{Bool} & \xrightarrow{\mathrm{init}} & \mathrm{Bit} \\ & \searrow^{\eta} & \downarrow{\scriptstyle \mathrm{dynlift}} \\ & & T\mathrm{Bool} \end{array}$$

  where $\psi : \mathcal{M} \to V(\mathcal{A})$ and $\phi : \mathcal{Q} \to V(\mathcal{A})_{VT}$ are strong monoidal embedding functors and ϕ preserves convex sum.

Concrete model constructed from biset ($\mathcal{V} = \mathbf{Set}^{2^{\mathrm{op}}}$) enriched category

Discussion and future work

Enriched category and the box-unbox operations $\flat(S \multimap U) \cong \mathcal{A}(S, U)$

Different shapes of computation:

- two-levels of compilation (Biset enriched category)
- branching structure from dynamic lifting
- recursive function and cycle

Computational cost of evaluation

Thank you for listening!