Semantics of a Quantum Programming Language

Peter Selinger

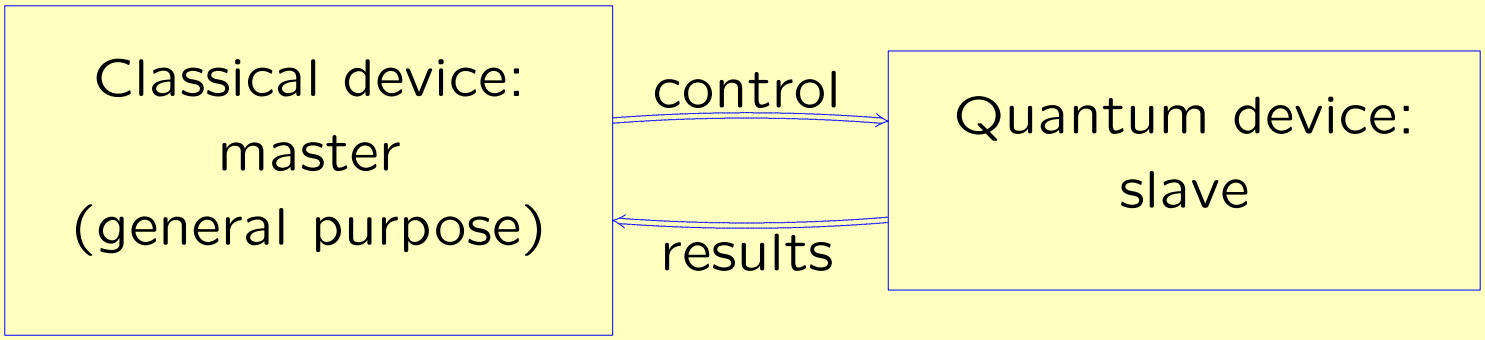
Dalhousie University
Halifax, Canada

Why Quantum Programming Languages?

- For certain problems, quantum algorithms have an exponential speedup over best known classical algorithms.
- Most research in quantum computing has focused on algorithms and complexity theory.
- Quantum algorithms are traditionally described in terms of hardware: quantum circuits or quantum Turing machines.
- Want compositionality. Also, how do quantum features interact with other language features such as structured data, recursion, i/o, higher-order.

Part I: Quantum Computation

The QRAM abstract machine [Knill96]



• General-purpose classical computer controls a special quantum hardware device

• Quantum device provides a bank of individually addressable qubits.

• Left-to-right: instructions.

• Right-to-left: results.

Linear Algebra Review

- Scalars $\lambda \in \mathbb{C}$, column vectors $u \in \mathbb{C}^n$, matrices $A \in \mathbb{C}^{n\times m}$.
- Adjoint $A^* = (\overline{a_{ji}})_{ij}$, trace $\operatorname{tr} A = \sum_i a_{ii}$, norm $\|A\|^2 = \sum_{ij} |a_{ij}|^2$.
- Unitary matrix $S \in \mathbb{C}^{n\times n}$ if $S^*S = I$.
  Change of basis: $B = SAS^* \Rightarrow \operatorname{tr} B = \operatorname{tr} A$, $\|B\| = \|A\|$.
- Hermitian matrix $A \in \mathbb{C}^{n\times n}$: if $A = A^*$.
  Hermitian positive: $u^*Au \geq 0$ for all $u \in \mathbb{C}^n$.
  Diagonalization: $A = SDS^*$, S unitary, D real diagonal.
- Tensor product $A \otimes B$, e.g. $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \otimes B = \begin{pmatrix} 0 & B \\ -B & 0 \end{pmatrix}$.

Quantum computation: States

- state of one qubit: $\alpha|\mathbf{0}\rangle + \beta|\mathbf{1}\rangle$ (*superposition* of $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$).
- state of two qubits: $\alpha|\mathbf{00}\rangle + \beta|\mathbf{01}\rangle + \gamma|\mathbf{10}\rangle + \delta|\mathbf{11}\rangle$.
- *independent*: $(a|\mathbf{0}\rangle + b|\mathbf{1}\rangle) \otimes (c|\mathbf{0}\rangle + d|\mathbf{1}\rangle) = ac|\mathbf{00}\rangle + ad|\mathbf{01}\rangle + bc|\mathbf{10}\rangle + bd|\mathbf{11}\rangle$.
- otherwise *entangled*.

Lexicographic convention

Identify the basis states $|\mathbf{00}\rangle$, $|\mathbf{01}\rangle$, $|\mathbf{10}\rangle$, $|\mathbf{11}\rangle$ with the standard basis vectors

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

in the *lexicographic* order.

Note: we use *column vectors* for states.

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix} = \alpha|\mathbf{00}\rangle + \beta|\mathbf{01}\rangle + \gamma|\mathbf{10}\rangle + \delta|\mathbf{11}\rangle.$$

Quantum computation: Operations

- unitary transformation

- measurement

Some standard unitary gates

Unary:

$$N = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

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

$$V = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix},$$

$$W = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{i} \end{pmatrix},$$

Binary:

$$N_c = \left(\begin{array}{c|c} I & 0 \\ \hline 0 & N \end{array}\right),$$

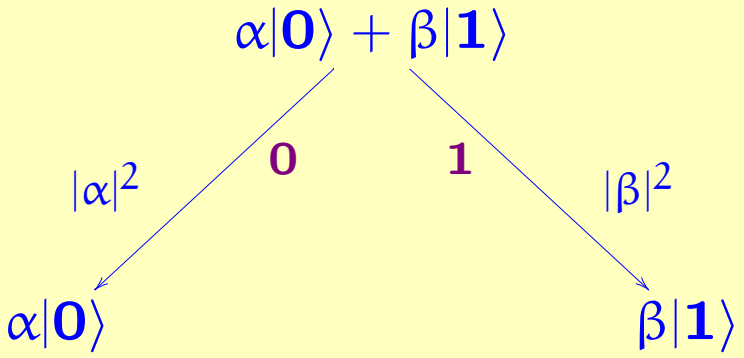
$$H_c = \left(\begin{array}{c|c} I & 0 \\ \hline 0 & H \end{array}\right),$$

$$V_c = \left(\begin{array}{c|c} I & 0 \\ \hline 0 & V \end{array}\right),$$

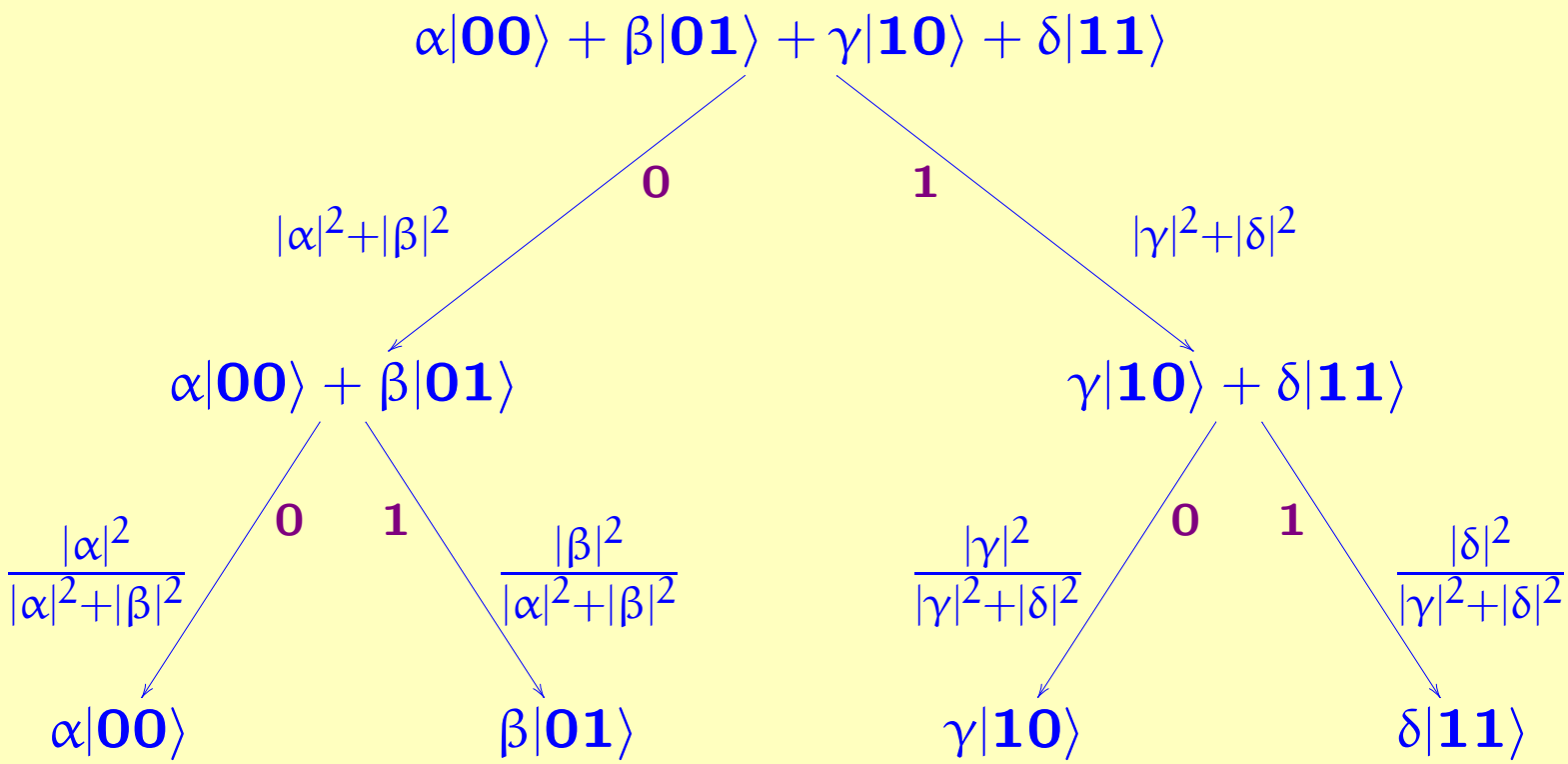
$$W_c = \left(\begin{array}{c|c} I & 0 \\ \hline 0 & W \end{array}\right),$$

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

Measurement



Two Measurements



Note: Normalization convention.

Pure vs. mixed states

A mixed state is a (classical) probability distribution on quantum states.

Ad hoc notation:

$$\frac{1}{2}\left\{\begin{pmatrix}\alpha\\ \beta\end{pmatrix}\right\}+\frac{1}{2}\left\{\begin{pmatrix}\alpha'\\ \beta'\end{pmatrix}\right\}$$

Note: A mixed state is a description of our *knowledge* of a state. An actual closed quantum system is always in a (possibly unknown) pure state.

Density matrices (von Neumann)

Represent the pure state $v = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \in \mathbb{C}^2$ by the matrix

$$vv^* = \begin{pmatrix} \alpha\bar{\alpha} & \alpha\bar{\beta} \\ \beta\bar{\alpha} & \beta\bar{\beta} \end{pmatrix} \in \mathbb{C}^{2\times 2}.$$

Represent the mixed state $\lambda_1\{v_1\} + \ldots + \lambda_n\{v_n\}$ by

$$\lambda_1 v_1 v_1^* + \ldots + \lambda_n v_n v_n^*.$$

This representation is not one-to-one, e.g.

$$\frac{1}{2}\left\{\begin{pmatrix} 1 \\ 0 \end{pmatrix}\right\} + \frac{1}{2}\left\{\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right\} = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} .5 & 0 \\ 0 & .5 \end{pmatrix}$$

$$\frac{1}{2}\left\{\frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\right\} + \frac{1}{2}\left\{\frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}\right\} = \frac{1}{2}\begin{pmatrix} .5 & .5 \\ .5 & .5 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} .5 & -.5 \\ -.5 & .5 \end{pmatrix} = \begin{pmatrix} .5 & 0 \\ 0 & .5 \end{pmatrix}$$

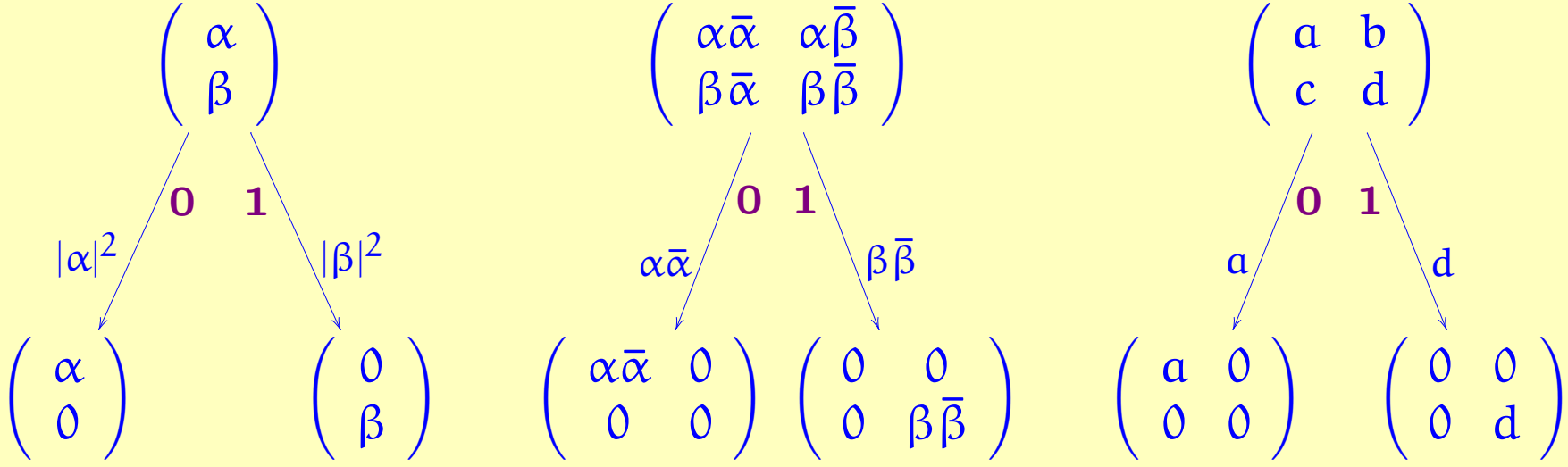
But these two mixed states are indistinguishable.

Quantum operations on density matrices

Unitary:

$$v \mapsto Uv \qquad vv^* \mapsto Uvv^*U^* \qquad A \mapsto UAU^*$$

Measurement:



A complete partial order of density matrices

Let $D_n = \{A \in \mathbb{C}^{n \times n} \mid A$ is positive hermitian and $\operatorname{tr} A \leq 1\}$.

Definition. We write $A \sqsubseteq B$ if $B - A$ is positive.

Theorem. The density matrices form a *complete partial order* under $\sqsubseteq$.

- $A \sqsubseteq A$
- $A \sqsubseteq B$ and $B \sqsubseteq A \Rightarrow A = B$
- $A \sqsubseteq B$ and $B \sqsubseteq C \Rightarrow A \sqsubseteq C$
- every increasing sequence $A_1 \sqsubseteq A_2 \sqsubseteq \ldots$ has a least upper bound

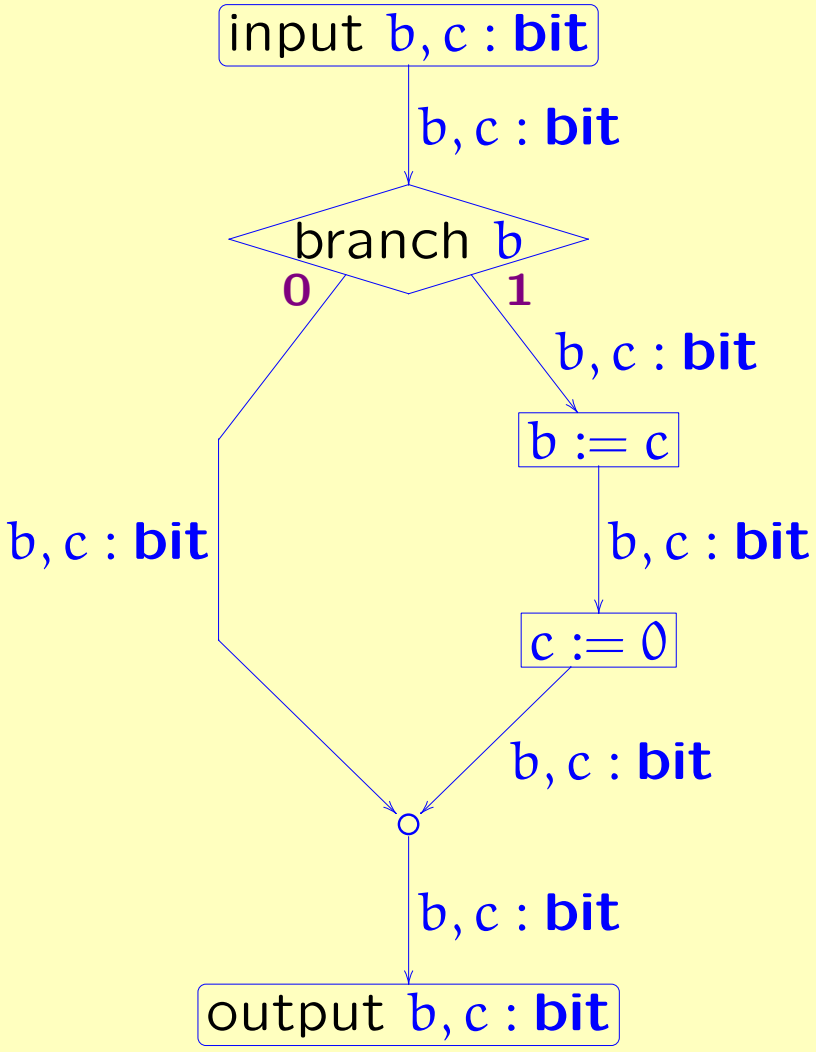
Part II: The Flow Chart Language

Earlier Quantum Programming Languages

- Knill (1996): conventions for writing pseudo-code
- Ömer (1998): scratch space management, user defined operators
- Sanders and Zuliani (2000): specification language, stepwise refinement
- Bettelli, Calarco, and Serafini (2001): based on C++

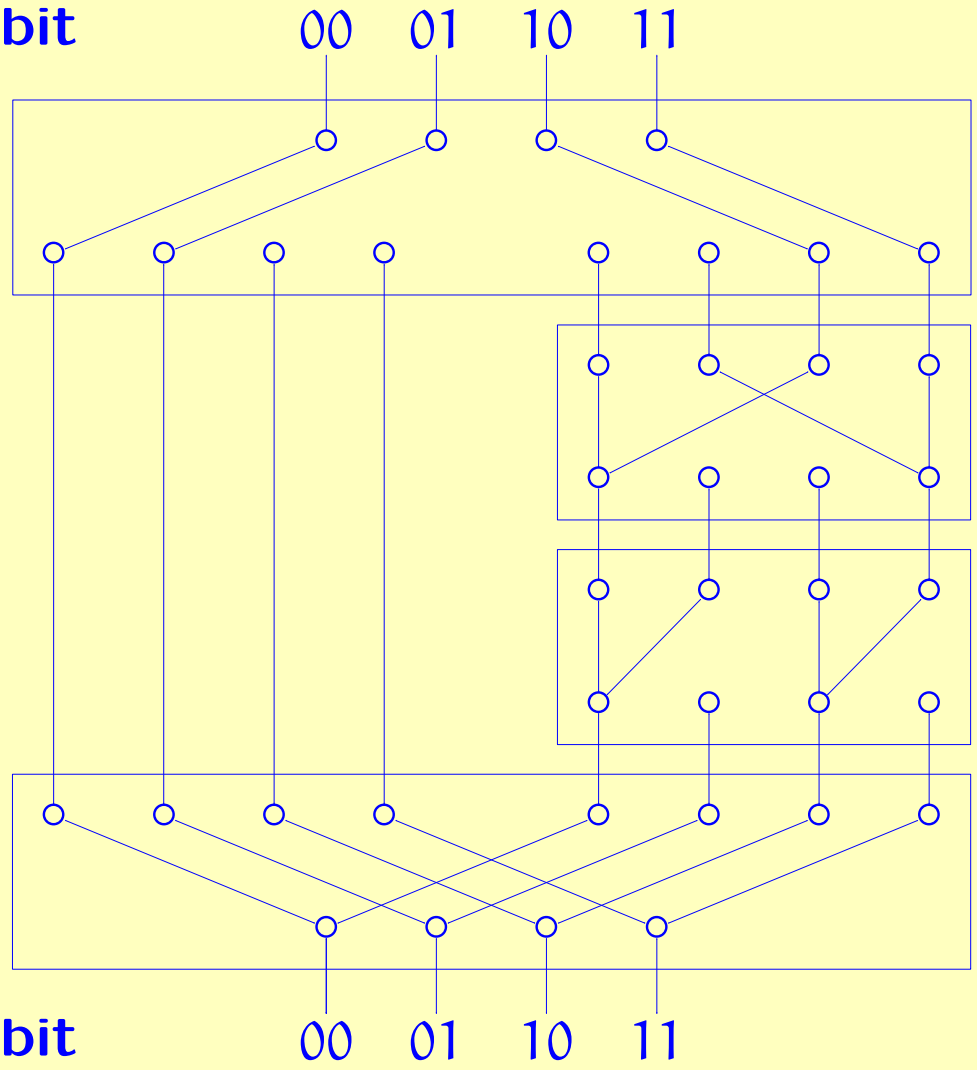
Imperative languages, run-time checks and errors, no formal semantics.

A simple classical flow chart



Classical flow chart, with boolean variables expanded

input b, c : **bit**



(∗ branch b ∗)

(∗ $b := c$ ∗)

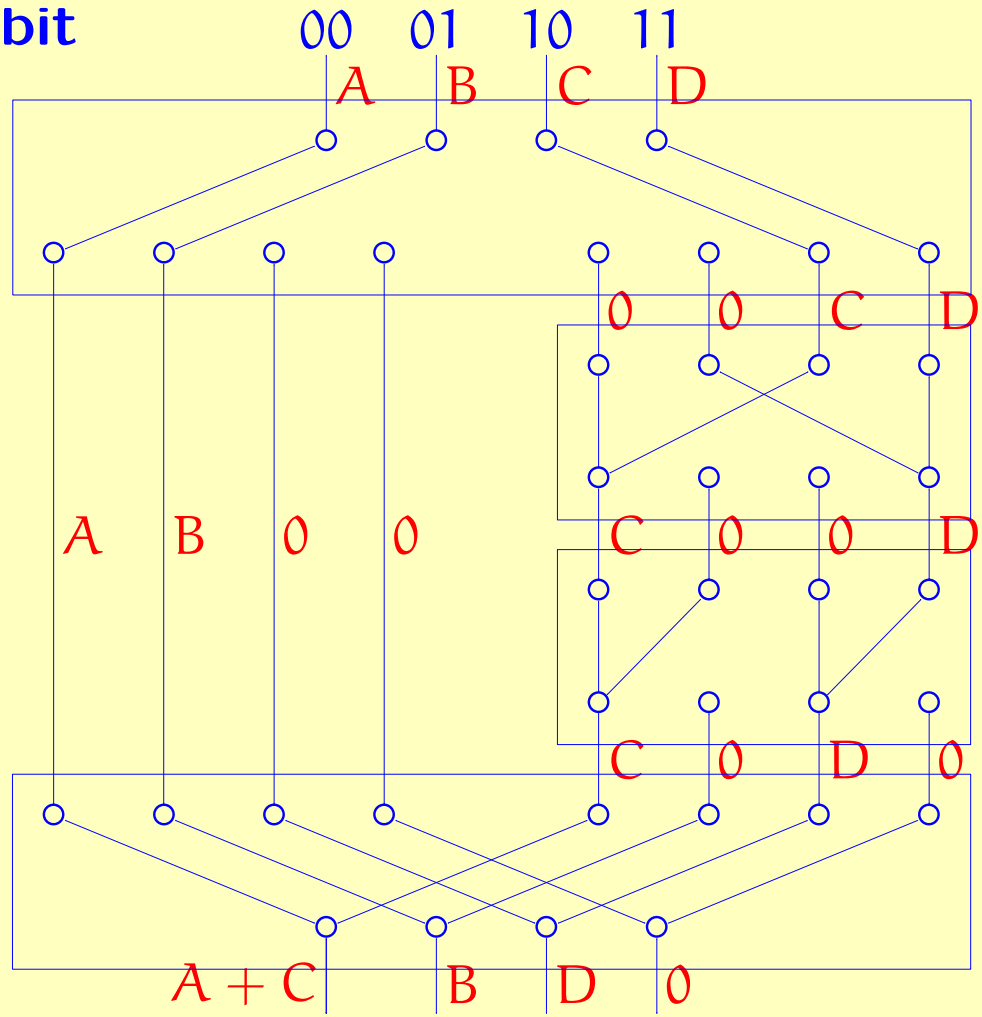
(∗ $c := 0$ ∗)

(∗ merge ∗)

output b, c : **bit**

Classical flow chart, with boolean variables expanded

input b, c : **bit**



output b, c : **bit**

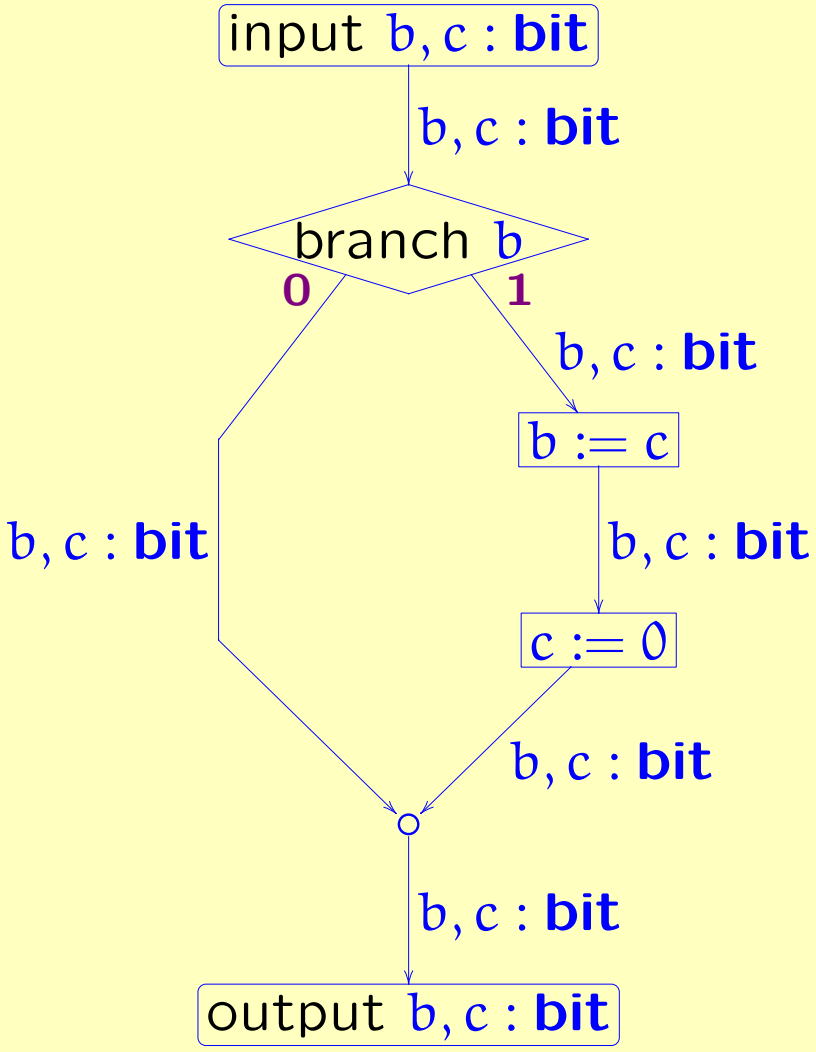
$(*$ branch b $*)$

$(*$ $b := c$ $*)$

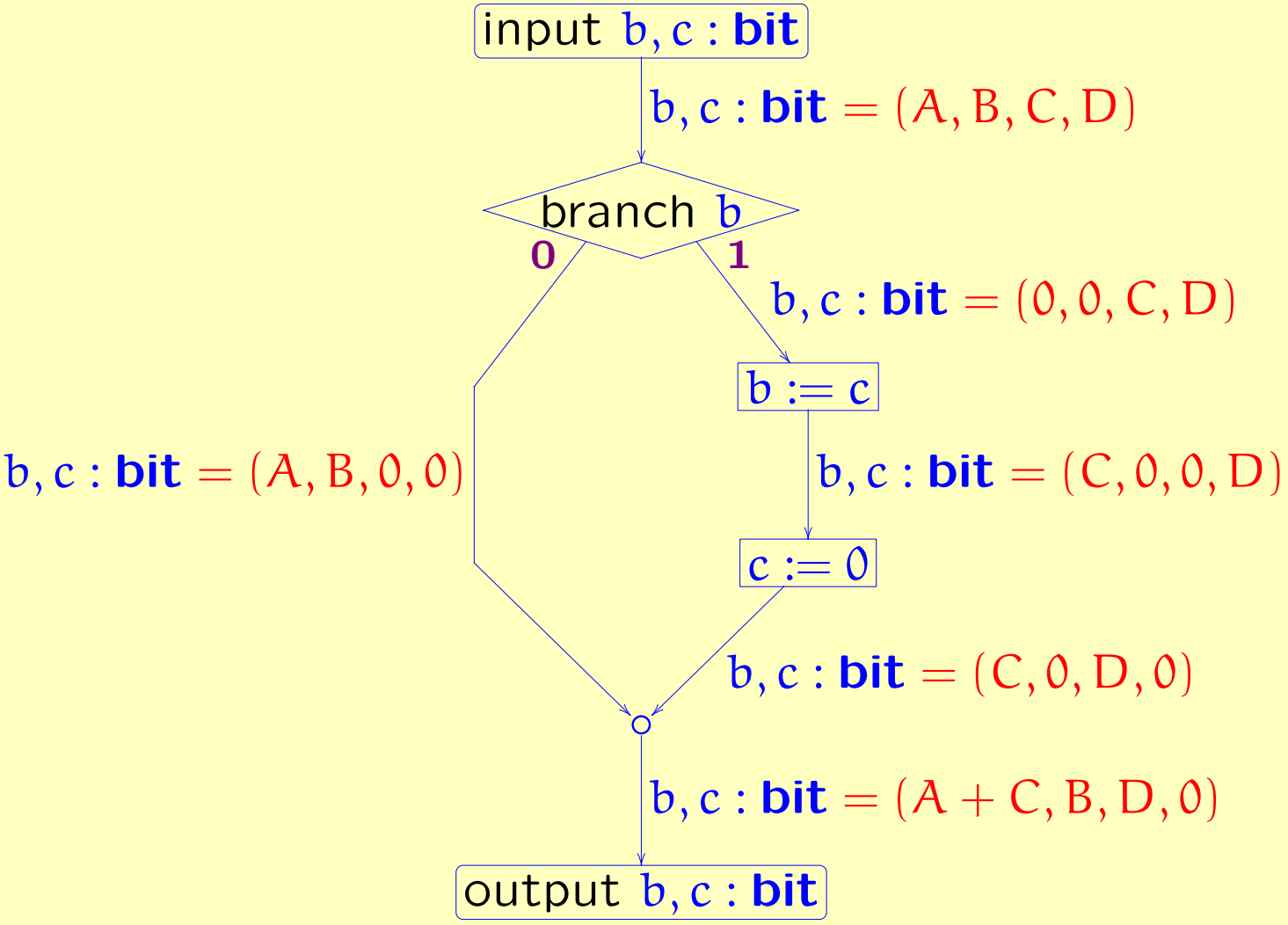
$(*$ $c := 0$ $*)$

$(*$ merge $*)$

A simple classical flow chart

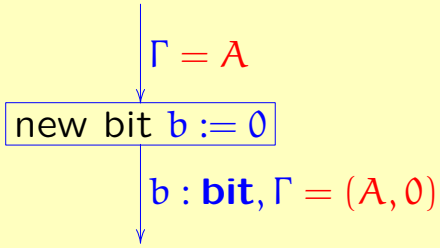


A simple classical flow chart

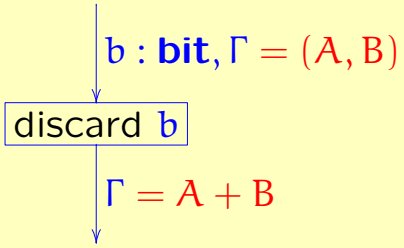


Summary of classical flow chart components

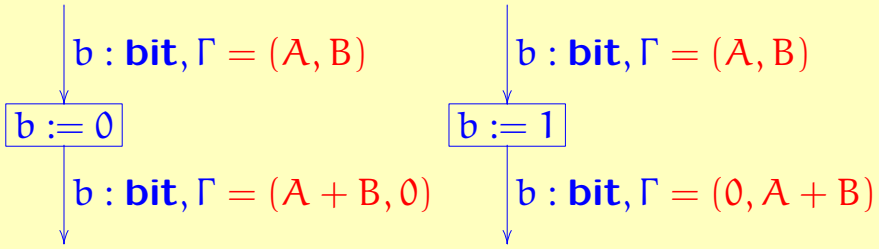
Allocate bit:



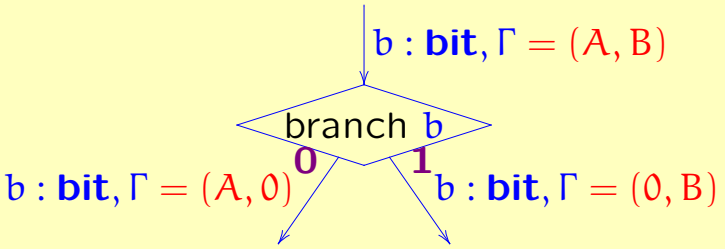
Discard bit:



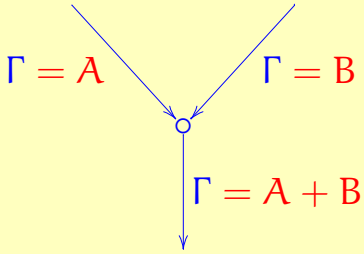
Assignment:



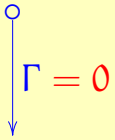
Branching:



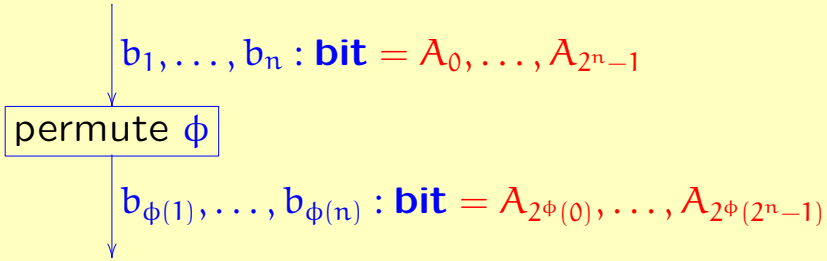
Merge:



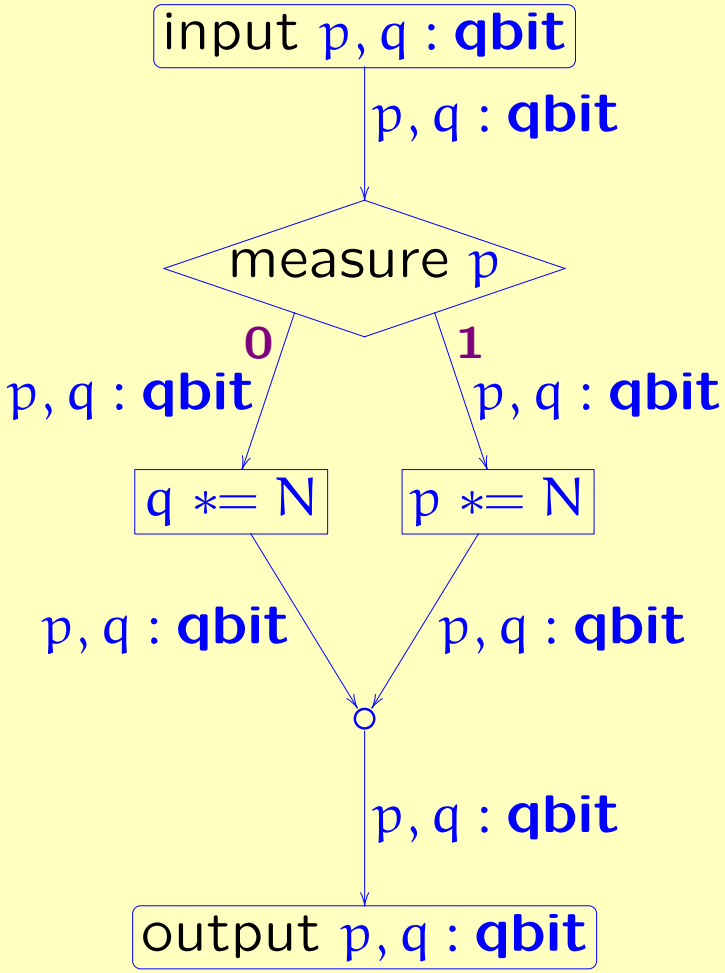
Initial:



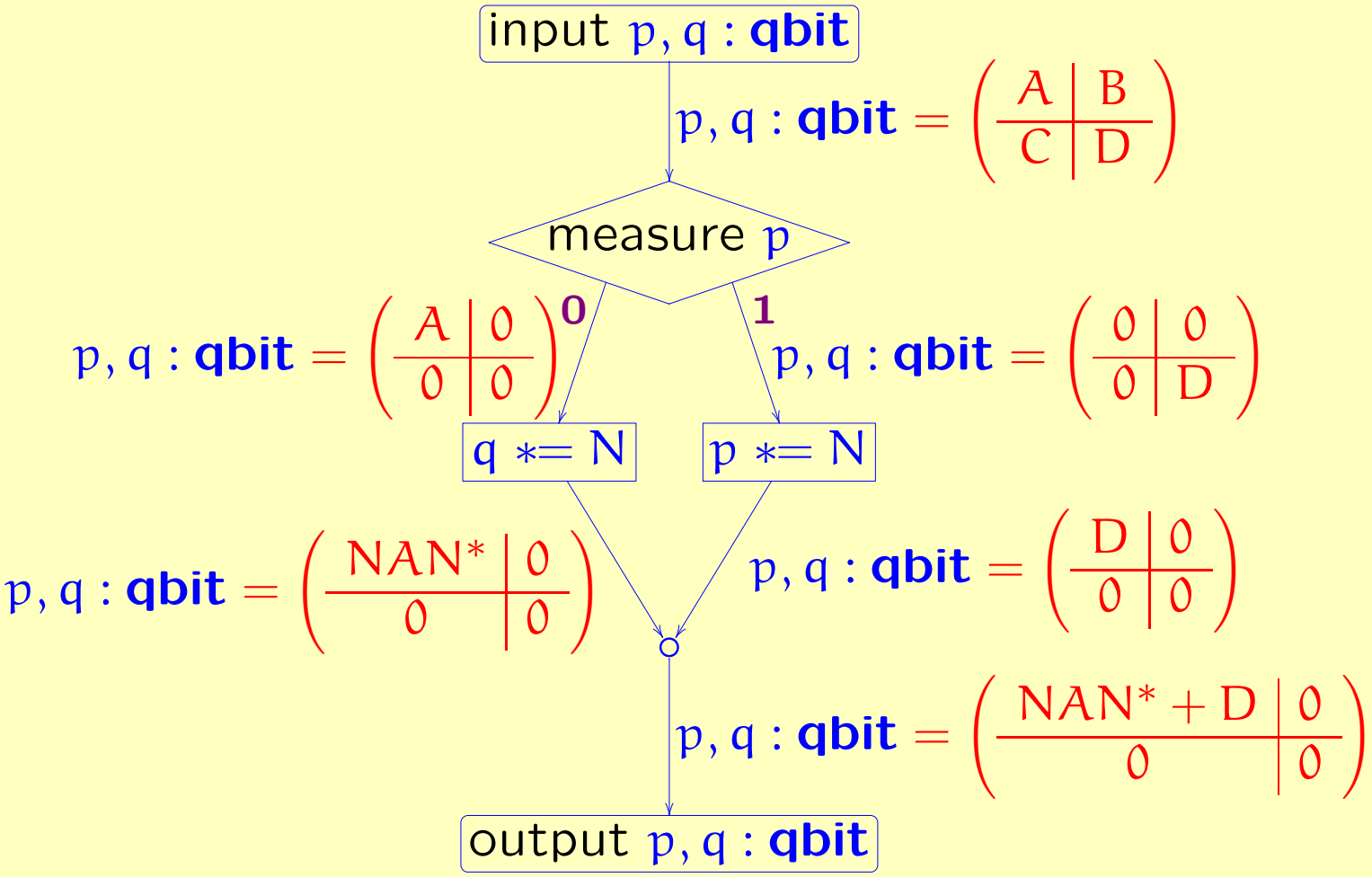
Permutation:



A simple quantum flow chart

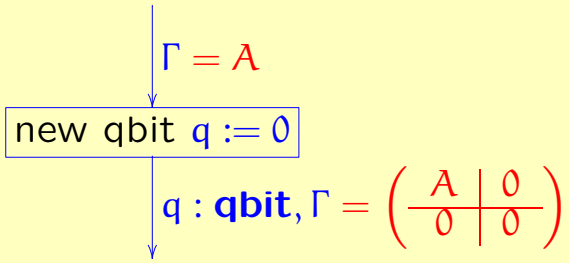


A simple quantum flow chart

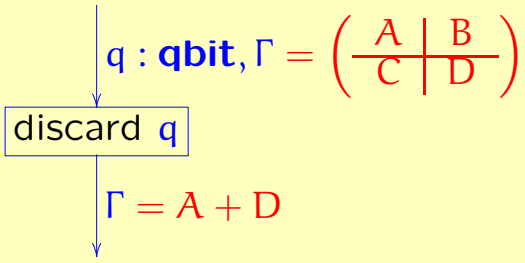


Summary of quantum flow chart components

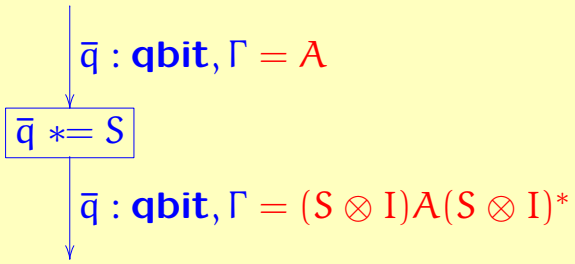
Allocate qbit:



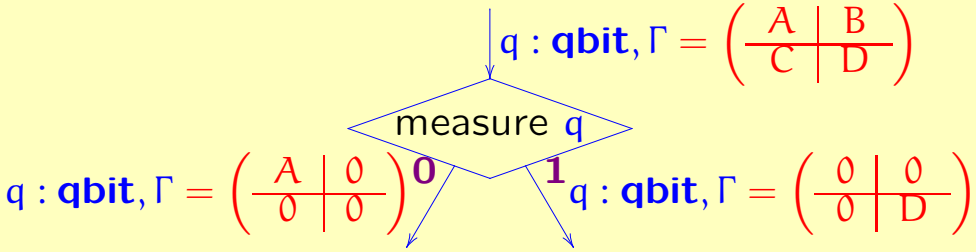
Discard qbit:



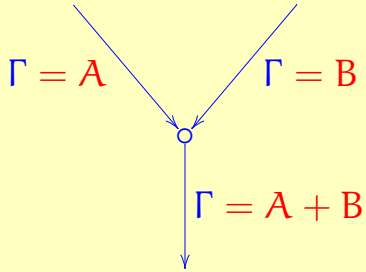
Unitary transformation:



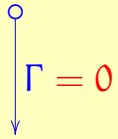
Measurement:



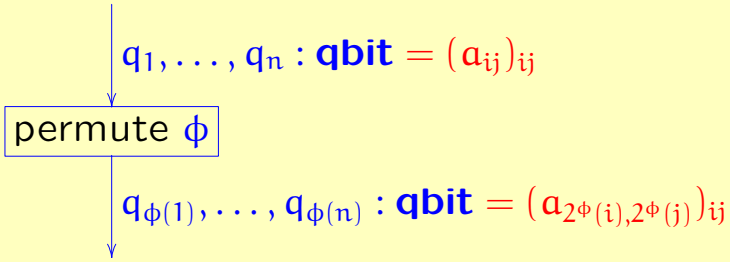
Merge:



Initial:



Permutation:



Combining classical data with quantum data

Consider typing contexts of the form

$$b_1 : \mathbf{bit}, \ldots, b_n : \mathbf{bit}, q_1 : \mathbf{qbit}, \ldots, q_m : \mathbf{qbit}.$$

Definition. A *state* for the above typing context is a 2^n-tuple $(A_0, \ldots, A_{2^n-1})$ of density matrices, each of dimension $2^m \times 2^m$.

$$\begin{array}{lcl} \mathrm{tr}\,(A_0, \ldots, A_{2^n-1}) & := & \sum_i \mathrm{tr}\, A_i, \\ (A_0, \ldots, A_{2^n-1})^* & := & (A_0^*, \ldots, A_{2^n-1}^*), \\ S(A_0, \ldots, A_{2^n-1})S^* & := & (SA_0S^*, \ldots, SA_{2^n-1}S^*), \\ |(A_0, \ldots, A_{2^n-1})|^2 & := & \sum_i |A_i|^2. \end{array}$$

Summary of language features:

- our language is *functional* (no side effects) and *statically typed* (no run-time errors).
- it combines *quantum and classical features* (the compiler can separate them again).
- it has *high-level features* (such as loops, recursion, and structured data types) [not shown in this talk]
- there is a *compositional denotational semantics*.

The denotation of a quantum flow chart

The denotation of a flow chart is a function which maps (tuples of) matrices to (tuples of) matrices.

Example: the denotation of the quantum flow chart from previous slide is the function

$$\mathbb{F}\left(\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right) = \left(\begin{array}{c|c} NAN^* + D & 0 \\ \hline 0 & 0 \end{array}\right).$$

Question: Which functions can occur?

Superoperators

1) *linear*

2) *positive:* A positive $\Rightarrow$ $\mathrm{F}(A)$ positive

3) *trace non-increasing:* A positive $\Rightarrow$ $\mathrm{tr}\,\mathrm{F}(A) \leq \mathrm{tr}(A)$

4) *completely positive:* $\mathrm{F} \otimes \mathrm{id}_n$ positive for all n

Theorem: The above conditions are necessary and sufficient.

The category $\mathbf{Q}$ of superoperators

Objects: signatures $\sigma = n_1, \ldots, n_k$

Morphisms: $f : \sigma \to \tau$ is a superoperator

$$f : \mathbb{C}^{n_1 \times n_1} \times \ldots \times \mathbb{C}^{n_k \times n_k} \to \mathbb{C}^{m_1 \times m_1} \times \ldots \times \mathbb{C}^{m_k \times m_{k'}}$$

Structure:

- symmetric monoidal category (horiz.+vert. composition)
- coproducts (merge, initial)
- CPO-enriched (fixpoints, recursion)
- traced monoidal (loops)

Structural and denotational equivalence

Definition. An *elementary quantum flow chart category* is a symmetric monoidal category with traced finite coproducts, such that $A \otimes (-)$ is a traced monoidal functor for every object A, together with a distinguished object **qbit** and morphisms $\nu : I \oplus I \to \mathbf{qbit}$ and $\mu : \mathbf{qbit} \to I \oplus I$, such that $\mu \circ \nu = \mathrm{id}$.

Definition. Two quantum flow charts X, Y are *structurally equivalent* if for every elementary quantum flow chart category $\mathbf{C}$ and every interpretation η of basic operator symbols, $[\![X]\!]_\eta = [\![Y]\!]_\eta$.

We say X and Y are *denotationally equivalent* if $[\![X]\!] = [\![Y]\!]$ for the canonical interpretation in the category $\mathbf{Q}$ of signatures and superoperators.

Higher-order quantum computation

- Consider functions of higher-order types such as $(\mathbf{qbit} \to \mathbf{bit}) \to \mathbf{qbit}$ etc.
- Quantum data is subject to *linearity constraints*. Need to avoid terms that lead to runtime errors such as

$$\mathsf{let}\ q = \mathsf{new\_qbit}()\ \mathsf{in}\ (\lambda x.\mathrm{H}(x, x))q.$$

- Bits are always duplicable, qubits are never duplicable. What about functions?
- Consider

$$\begin{array}{lcll} q{:}\mathbf{qbit} & \vdash & \lambda p.p & : \ \mathbf{qbit} \to \mathbf{qbit} \\ q{:}\mathbf{qbit} & \vdash & \lambda p.q & : \ \mathbf{qbit} \to \mathbf{qbit} \end{array}$$

  Both closures have type $\mathbf{qbit} \to \mathbf{qbit}$, but only the first one is duplicable.

Linear type system [Selinger,Valiron04]

Types: $A, B ::= \alpha \mid !A \mid A \multimap B \mid 1 \mid A \otimes B$.

Subtyping: $!A <: A$.

Typing rules:

$$\frac{x{:}A, \Delta \vdash M : B}{\Delta \vdash \lambda x.M : A \multimap B}\,(\lambda_1) \qquad \text{If } FV(M) \cap |\Gamma| = \emptyset\text{:} \quad \frac{\Gamma, !\,\Delta, x{:}A \vdash M : B}{\Gamma, !\,\Delta \vdash \lambda x.M : !(A \multimap B)}\,(\lambda_2)$$

(Operational semantics, subject reduction, progress, safety, type inference).

Example: quantum teleportation / dense coding

It is possible to write a function

$$\mathbb{TPPair} : 1 \to (\mathbf{qbit} \multimap \mathbf{bit} \otimes \mathbf{bit}) \otimes (\mathbf{bit} \otimes \mathbf{bit} \multimap \mathbf{qbit})$$

such that for any application

$$(\mathrm{f}, \mathrm{g}) = \mathbb{TPPair}(),$$

one obtains a pair of functions (f, g) with the properties:

$$\mathrm{f} : \mathbf{qbit} \multimap \mathbf{bit} \otimes \mathbf{bit}$$
$$\mathrm{g} : \mathbf{bit} \otimes \mathbf{bit} \multimap \mathbf{qbit}$$

$$\mathrm{f} \circ \mathrm{g} = \mathrm{id}_{\mathbf{bit} \otimes \mathbf{bit}}$$
$$\mathrm{g} \circ \mathrm{f} = \mathrm{id}_{\mathbf{qbit}}$$

Thus, are the types **qbit** and $\mathbf{bit} \otimes \mathbf{bit}$ isomorphic?

Answer: No, *because each such pair* f, g *can only be used once.* Thus we have a "single-use type isomorphism". This is a curious phenomenon.

Overview of some recent research

- **Quantum process calculi.** Lalire-Jorrand (2004), Gay-Nagarajan (2004), Adão-Mateus (2005)
- **Higher-order quantum computation.** Van Tonder (2003, 2004), Selinger-Valiron (2004), Altenkirch-Grattage (2004)
- **Categorical quantum computation.** Abramsky-Coecke (2004), Selinger (2005)
- **Measurement based quantum computation.** Danos-D'Hondt-Kashefi-Panangaden (2004, 2005)
- **Quantum specification.** Zuliani (2001-2004), D'Hondt-Panangaden (2004), Tafliovich (2004)
- **Quantum coherent spaces.** Girard (2003), Selinger (2004)