A Cook's tour of a simple quantum programming language

Samson Abramsky
Oxford University Computing Laboratory

Motivation

- A **simple** language.
- Several different semantics, correspondences between them.
- Simple as it is, doesn't seem to have been done before.
- Incremental method: build up from deterministic to probabilistic to quantum.

While Programs

Given some syntax of **arithmetic expressions** aexp, **boolean expressions** bexp, and **program variables** var, we define the syntax of **commands**.

$$\begin{array}{rcl} C & ::= & \mathsf{var} := \mathsf{aexp} \\ & & \mid \mathsf{skip} \\ & & \mid C_1 ; C_2 \\ & & \mid \mathsf{if\ bexp\ then}\ C_1\ \mathsf{else}\ C_2 \\ & & \mid \mathsf{while\ bexp\ do}\ C. \end{array}$$

Structural Operational Semantics

Configurations are pairs (C, s) where C is a command and s is a **state**, *i.e.* a mapping from variables to values. We write $\mathcal{S}$ for the set of states.

Configurations of the form (skip, s) are **terminal**.

We define a one-step transition relation $\rightarrow$ between configurations, by structural induction on the command component.

The transition relation

$$\frac{}{(v := e, s) \to (\mathsf{skip}, s[v \mapsto [\![e]\!]s])}$$

$$\frac{}{(\mathsf{skip}; C, s) \to (C, s)} \qquad \frac{(C_1, s) \to (C_1', s')}{(C_1; C_2, s) \to (C_1'; C_2, s')}$$

$$\frac{[\![b]\!]s = \mathtt{tt}}{(\mathsf{if}\ b\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, s) \to (C_1, s)} \qquad \frac{[\![b]\!]s = \mathtt{ff}}{(\mathsf{if}\ b\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, s) \to (C_2, s)}$$

$$\frac{[\![b]\!]s = \mathtt{tt}}{(\mathsf{while}\ b\ \mathsf{do}\ C, s) \to (C; \mathsf{while}\ b\ \mathsf{do}\ C, s)} \qquad \frac{[\![b]\!]s = \mathtt{ff}}{(\mathsf{while}\ b\ \mathsf{do}\ C, s) \to (\mathsf{skip}, s)}$$

This semantics is completely **deterministic**. The transition relation $\to$ is a **partial function** on configurations (and only undefined on (C, s) when $C = \mathsf{skip}$).

The SOS gives rise to a meaning function

$$\mathcal{O}[\![C]\!] : \mathcal{S} \rightharpoonup \mathcal{S}$$

defined by

$$\mathcal{O}[\![C]\!](s) = \begin{cases} s', & \text{if } (C, s) \to^* (\mathsf{skip}, s') \\ \text{undefined} & \text{otherwise} \end{cases}$$

Denotational Semantics

$$\mathcal{D}[\![C]\!] : \mathcal{S} \rightharpoonup \mathcal{S}$$

The denotational semantics defines this 'global' denotation **compositionally**.

$$
\begin{array}{lcl}
\mathcal{D}[\![\mathsf{skip}]\!] & = & \mathsf{id}_{\mathcal{S}} \\
\mathcal{D}[\![v := e]\!](s) & = & s[v \mapsto [\![e]\!]s]) \\
\mathcal{D}[\![C_1 ; C_2]\!] & = & \mathcal{D}[\![C_2]\!] \circ \mathcal{D}[\![C_1]\!] \\
\mathcal{D}[\![\mathsf{if}\ b\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2]\!] & = & \text{if } [\![b]\!]s = \mathtt{tt} \text{ then } \mathcal{D}[\![C_1]\!](s) \text{ else } \mathcal{D}[\![C_2]\!](s) \\
\mathcal{D}[\![\mathsf{while}\ b\ \mathsf{do}\ C]\!] & = & \mathbf{Y}(\lambda f : \mathcal{S} \rightharpoonup \mathcal{S}.\, \lambda s : \mathcal{S}. \\
 & & \text{if } [\![b]\!]s = \mathtt{tt} \text{ then } (f \circ \mathcal{D}[\![C]\!])(s) \text{ else } s)
\end{array}
$$

Here $\mathbf{Y} : [[\mathcal{S} \rightharpoonup \mathcal{S}] \rightharpoonup [\mathcal{S} \rightharpoonup \mathcal{S}]] \rightharpoonup [\mathcal{S} \rightharpoonup \mathcal{S}]$. We are using the fact that $[\mathcal{S} \rightharpoonup \mathcal{S}]$ is a **domain**, and all meanings are **continuous**, so the least-fixed-point theorem applies.

Correspondence Theorem

For all commands C:

$$\mathcal{O}[\![C]\!] = \mathcal{D}[\![C]\!].$$

Dynamic Logic

We define a set of formulas.

$$\phi \quad ::= \quad \top \mid \phi \wedge \psi \mid \neg\phi \mid \langle C\rangle\phi.$$

Semantics: $\|\phi\| \subseteq \mathcal{S}$.

$$\begin{array}{lcl}
\|\top\| & = & \mathcal{S} \\
\|\phi \wedge \psi\| & = & \|\phi\| \cap \|\psi\| \\
\|\neg\phi\| & = & \mathcal{S} \setminus \|\phi\| \\
\|\langle C\rangle\phi\| & = & \{s \in \mathcal{S} \mid \mathcal{D}[\![C]\!](s) \in \|\phi\|\}
\end{array}$$

Logical semantics

Weakest precondition semantics: $\mathrm{WP}[\![C]\!] : \mathcal{P}(\mathcal{S}) \longrightarrow \mathcal{P}(\mathcal{S})$.

$$\mathrm{WP}[\![C]\!](U) = \langle C \rangle (U).$$

This can be shown equivalent to our other forms of semantics. Also, the set of formulas true for a command characterize it up to equivalence wrt. the other forms of semantics.

A Probabilistic Language

We extend the basic language with one new construct.

$$C \quad ::= \quad \ldots \mid \mathsf{cointoss}\ p\ \mathsf{in}\ v$$

Here $0 \leq p \leq 1$. Intuitively, we toss a coin with probability p for heads, $1 - p$ for Tails. If the coin lands on Heads, we assign 0 to v, and if it lands Tails we assign 1.

Operational Semantics

Now transitions are labelled with probabilities such that, for all non-terminal configurations (C, s):

$$\sum \{p \mid (C, s) \xrightarrow{p} (C', s')\} = 1.$$

All previous transitions still apply, now labelled with 1.

New rules:

$$\frac{}{(\mathsf{cointoss}\ p\ \mathsf{in}\ v, s) \xrightarrow{p} (\mathsf{skip}, s[v \mapsto 0])} \qquad \frac{}{(\mathsf{cointoss}\ p\ \mathsf{in}\ v, s) \xrightarrow{1-p} (\mathsf{skip}, s[v \mapsto 1])}$$

The Semantic Function

We define a new space of **probabilistic state transformers**.

$\mathsf{DProb}(\mathcal{S})$ is the set of discrete sub-probability distributions on $\mathcal{S}$, *i.e.* functions $\mu : \mathcal{S} \to [0,1]$ such that

$$\sum_{s \in \mathcal{S}} \mu(s) \leq 1.$$

Program meanings have the form $\mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$. We define $\mathsf{Comp}(C, s, s')$ to be the set of all sequences

$$(C, s) = (C_0, s_0) \xrightarrow{p_1} \cdots \xrightarrow{p_k} (C_k, s_k) = (\mathsf{skip}, s').$$

If c is such a sequence, we define $p(c)$ to be the product $\prod_{i=1}^{k} p_i$. Then

$$\mathcal{O}[\![C]\!](s)(s') = \sum \{p(c) \mid c \in \mathsf{Comp}(C, s, s')\}.$$

The Domain of probabilistic state transformers

We use the pointwise ordering on probability distributions, and then on state transformers. This makes $\mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$ into a (continuous) domain.

Any partial function $f : \mathcal{S} \rightharpoonup \mathcal{S}$ can be embedded in $\mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$:

$$\hat{f}(s)(s') = \begin{cases} 1, & f(s) = s' \\ 0 & \text{otherwise} \end{cases}$$

Denotational Semantics

$$\mathcal{D}[\![\mathsf{skip}]\!] = \widehat{\mathcal{D}[\![\mathsf{skip}]\!]}$$

$$\mathcal{D}[\![v := e]\!] = \widehat{\mathcal{D}[\![v := e]\!]}$$

$$\mathcal{D}[\![\mathsf{cointoss}\ p\ \mathsf{in}\ v]\!](s)(s') = \begin{cases} p, & s' = s[v \mapsto 0] \\ 1-p, & s' = s[v \mapsto 1] \\ 0 & \text{otherwise} \end{cases}$$

Composition

$$f, g : \mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$$

$$f; g(s, s'') = \sum_{s' \in \mathcal{S}} f(s, s') \cdot g(s', s'')$$

Conditional

$$B : \mathcal{S} \to \{\mathtt{tt}, \mathtt{ff}\} \qquad f, g : \mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$$

$$\mathsf{cond}(B, f, g)(s) = \begin{cases} f(s), & B(s) = \mathtt{tt} \\ g(s) & B(s) = \mathtt{ff} \end{cases}$$

While

$$B : \mathcal{S} \to \{\mathtt{tt}, \mathtt{ff}\} \qquad f : \mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$$

$$\begin{aligned} \mathsf{while}(B, f) \quad = \quad & \mathbf{Y}(\lambda g : \mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S}).\, \lambda s : \mathcal{S}. \\ & \text{if } B(s) = \mathtt{tt} \text{ then } (f; g)(s) \text{ else } \widehat{\mathsf{id}}(s)). \end{aligned}$$

Once again we get a **Correspondence Theorem**:

$$\mathcal{O}[\![\cdot]\!] = \mathcal{D}[\![\cdot]\!].$$

A Probabilistic Dynamic Logic

$$\phi \quad ::= \quad \ldots \mid \langle C \rangle_q \phi$$

Here q is a rational number.

Semantics:

$$\|\langle C \rangle_q \phi\| = \{s \in \mathcal{S} \mid \mathbf{P}(\mathcal{D}[\![C]\!](s) \in \|\phi\|) \geq q\}$$

Here, if $f : \mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S})$ and $U \subseteq \mathcal{S}$, then

$$\mathbf{P}(f(s) \in U) = \sum_{s' \in U} f(s)(s').$$

Qubits

Not values (e.g. Booleans) but **program variables**.

Formally, the states of a qubit are 2-dimensional complex vectors of unit norm:

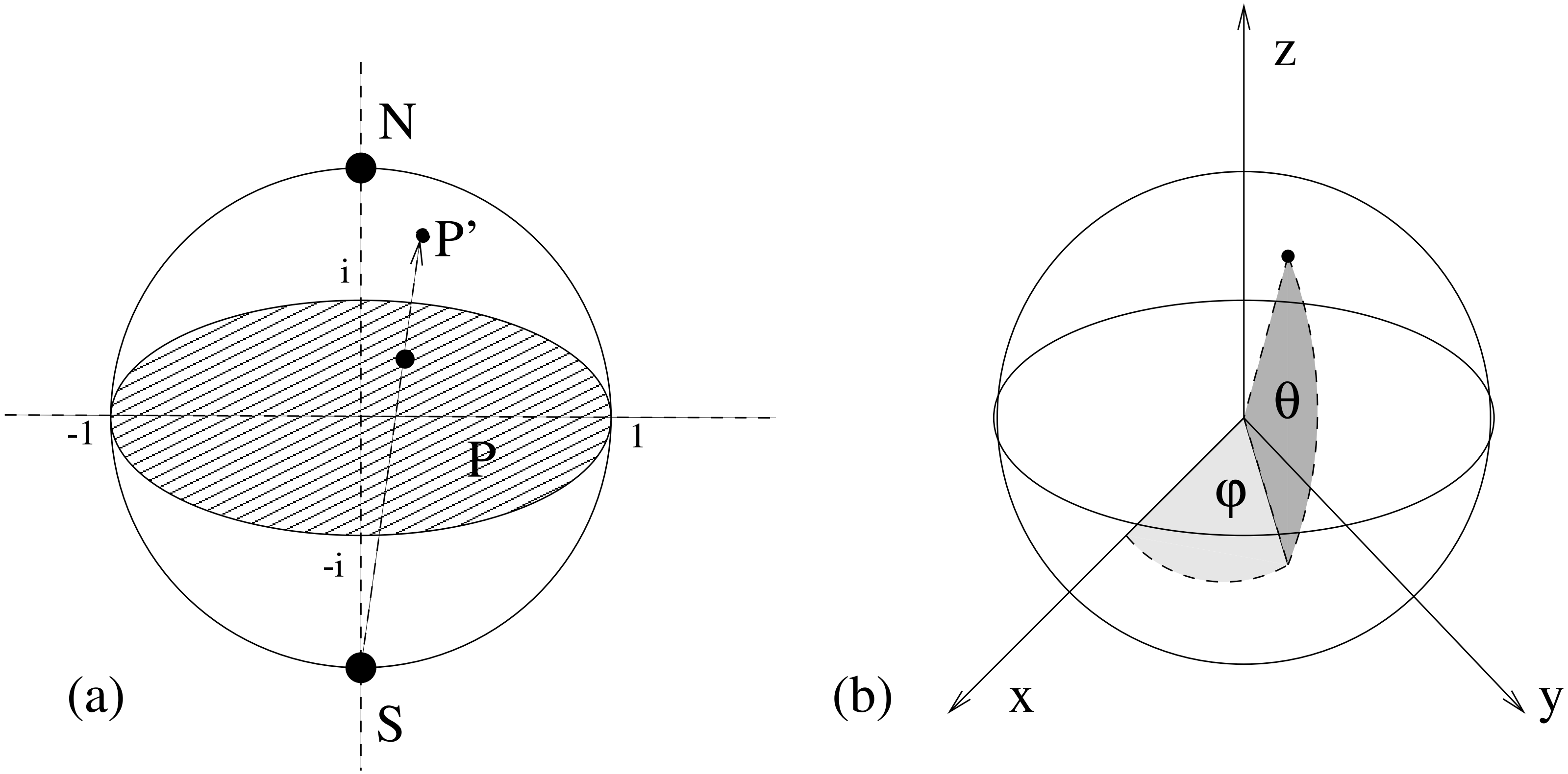
$$\alpha|0\rangle + \beta|1\rangle \qquad\qquad |\alpha|^2 + |\beta|^2 = 1.$$

A 'block' of k qubits is represented by a tensor product. The basis vectors are $|b\rangle$, $b \in \{0,1\}^k$. The general form of the state of such a block of qubits is:

$$\sum_{b\in\{0,1\}^k} \alpha_b|b\rangle \qquad\qquad \sum |\alpha_b|^2 = 1.$$

Note that such a state **cannot** be decomposed into a k-tuple of states of the components. The sum may encode **correlations** between the components, and hence **entanglement**.

Geometric visualizations of a single qubit



The Riemann sphere and the Bloch sphere

Operations on qubits

There are just two kinds.

- **Unitary operations** (norm preserving linear isomorphisms). There are 'complete' finite bases of unitaries consisting only of binary and unary operations. Unitary operations transform the quantum state in a purely deterministic fashion.

- **Measurements**. We only need to consider measurements in the computational basis, since measurements in other bases can be realized by first applying a unitary to transform the basis to the standard one.

  If we measure qubit i, $1 \leq i \leq k$, in a given state

  $$\sum_{b \in \{0,1\}^k} \alpha_b |b\rangle$$

  then one of two things can happen:

– with probability $p_0 = \sum\{|\alpha_b|^2 : b_i = 0\}$ we get the result 0, and the state collapses to

$$\frac{1}{\sqrt{p_0}} \sum\{\alpha_b|b\rangle : b_i = 0\}$$

– with probability $p_1 = \sum\{|\alpha_b|^2 : b_i = 1\}$ we get the result 1, and the state collapses to

$$\frac{1}{\sqrt{p_1}} \sum\{\alpha_b|b\rangle : b_i = 1\}.$$

Unitary operations

Suppose we have a quantum state consisting of k qubits. The state space is

$$Q^k = \underbrace{Q \otimes \cdots \otimes Q}_{k}.$$

To apply the binary unitary operation

$$U : Q \otimes Q \longrightarrow Q \otimes Q$$

to the i and j components of Q^k, we define $U_{i,j}$ to be the following composition:

Unitary operations continued

$$
\begin{tikzcd}
Q \otimes \cdots \otimes Q_i \otimes \cdots \otimes Q_j \otimes \cdots \otimes Q \arrow[d, "\sigma"] \\
(Q_i \otimes Q_j) \otimes Q^{k-2} \arrow[d, "U \otimes I"] \\
(Q_i \otimes Q_j) \otimes Q^{k-2} \arrow[d, "\sigma^{-1}"] \\
Q \otimes \cdots \otimes Q_i \otimes \cdots \otimes Q_j \otimes \cdots \otimes Q
\end{tikzcd}
$$

Measurements

For each quantum state $\phi \in Q^k$,

$$\phi = \sum_{b \in \{0,1\}^k} \alpha_b |b\rangle,$$

$1 \leq i \leq k$, and $j \in \{0,1\}$, we define:

$$p^i_j(\phi) = \sum \{|\alpha_b|^2 : b_i = j\}$$

$$P^i_j(\phi) = \frac{1}{\sqrt{p^i_j(\phi)}} \sum \{\alpha_b |b\rangle : b_i = j\}$$

Thus $p^i_j(\phi)$ is the probability that we will get the result j on measuring the i'th qubit in state ϕ, and $P^i_j(\phi)$ is the collapsed state that will result from such a measurement outcome.

A Simple Quantum Programming Language

We add the following constructs to our simple (deterministic or probabilistic) language. Firstly, in addition to our standard, 'classical' program variables, we also have some qubit variables q, q', q_1 etc.

For each of our basic stock of unitary operations U, we will have a command of the form $U(q_i, q_j)$, where q_i and q_j are qubit variables. (We can represent unary operations by binary ones which are the identity in one argument, so we only show the binary case).

Also, we have a command of the form

$$\mathsf{measure}\ q\ \mathsf{in}\ v$$

where q is a qubit variable, and v is a classical variable. The idea is that a measurement is performed on q, and the result (0 or 1) is stored in v.

Operational Semantics

Suppose we consider a program with k qubits. Configurations now have the form (C, s, ϕ), where (C, s) are as before, and $\phi \in Q^k$.

The transitions from the probabilistic language carry over essentially unchanged. For each transition

$$(C, s) \xrightarrow{p} (C', s')$$

in the probabilistic language, we have a transition

$$(C, s, \phi) \xrightarrow{p} (C', s', \phi)$$

in the quantum language. (We simply rewrite the transition rules with the quantum state ϕ carried through as a 'silent passenger').

So we just have to write rules for the new constructs.

Rules for the new constructs

$$\frac{}{(U(q_i, q_j), s, \phi) \xrightarrow{1} (\mathsf{skip}, s, U_{i,j}(\phi))}$$

$$\frac{}{(\mathsf{measure}\ q_i\ \mathsf{in}\ v, s, \phi) \xrightarrow{p_0^i(\phi)} (\mathsf{skip}, s[v \mapsto 0], P_0^i(\phi))}$$

$$\frac{}{(\mathsf{measure}\ q_i\ \mathsf{in}\ v, s, \phi) \xrightarrow{p_1^i(\phi)} (\mathsf{skip}, s[v \mapsto 1], P_1^i(\phi))}$$

The space of meanings

Just as before, we are going to assign meanings to commands as probabilistic stransformers. The process of measurement transforms **quantum superpositions with complex amplitudes** into **real-valued probabilistic branching**, so at the top level, our semantics is 'just' a standard Markov process.

A technical subtlety

The set of classical states $\mathcal{S}$ can always be taken to be countable. The set of quantum states Q^k, and hence also $\mathcal{S} \times Q^k$, cannot. So we define $\mathsf{DProb}(\mathcal{S} \times Q^k)$ to be the set of discrete sub-probability distributions **with countable support**, *i.e.* the set of states on which the distribution is non-zero is countable.

Note that, from the operational semantics, which is a finitely-branching Markov process, the set of possible final states (terminal configurations) for any given inital configuration is countable.

Applying the pointwise order to $\mathsf{DProb}(\mathcal{S} \times Q^k)$, and thence to

$$\mathcal{S} \times Q^k \longrightarrow \mathsf{DProb}(\mathcal{S} \times Q^k)$$

we note that these are ω-cpo's, but **not** dcpo's!

Meaning functions

We can define the operational meaning exactly as before (with the obvious adjustment of the set of computations

$$\mathsf{Comp}(C, (s, \phi), (s', \phi'))$$

to allow for the presence of the quantum state).

Moreover, the denotational semantics carries over essentially unchanged, modulo the fact that our semantic domain is now

$$\mathcal{S} \times Q^k \longrightarrow \mathsf{DProb}(\mathcal{S} \times Q^k)$$

rather than

$$\mathcal{S} \longrightarrow \mathsf{DProb}(\mathcal{S}).$$

We just have to write clauses for the new constructs.

Denotational semantics

$$\mathcal{D}[\![U(q_i, q_j]\!](s,\phi)(s',\psi) = \begin{cases} 1, & s' = s \ \wedge \ \psi = U_{i,j}(\phi) \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{D}[\![\mathsf{measure}\ q_i\ \mathsf{in}\ v]\!](s,\phi)(s',\psi) = \begin{cases} p_0^i(\phi), & s' = s[v \mapsto 0] \ \wedge \ \psi = P_0^i(\phi) \\ p_1^i(\phi), & s' = s[v \mapsto 1] \ \wedge \ \psi = P_1^i(\phi) \\ 0 & \text{otherwise} \end{cases}$$

Once again, we obtain a

Correspondence Theorem: $\mathcal{O}[\![\cdot]\!] = \mathcal{D}[\![\cdot]\!]$.

Logical Sematics (?)

Our probabilistic modality $\langle C \rangle_q$ still makes good sense for the quantum programming language.

However ...

New Question

What is a good logic for talking about the quantum part of the state, and how to make it fit smoothly with the rest of our logic?

Recent work by Alexandru Baltag and Sonja Smets is highly relevant.

A Monad of Quantum States

Define

$$T_k(X) \quad = \quad Q^k \Rightarrow \mathsf{DProb}(X \times Q^k)$$

Then

$$[\mathcal{S} \Rightarrow T_k(\mathcal{S})] \quad \cong \quad [\mathcal{S} \times Q^k \Rightarrow \mathsf{DProb}(\mathcal{S} \times Q^k)]$$

Unitaries and measurements as 'generic effects':

$$U : \mathbf{1} \longrightarrow T_k(\mathbf{1})$$

$$M^i : \mathbf{1} \longrightarrow T_k(\mathbf{1} + \mathbf{1})$$

A construction for general domains

To give a more general notion of quantum states on a Hilbert space $\mathcal{H}$, define $\mathbf{D}(\mathcal{H})$ to be the **density operators** on $\mathcal{H}$. We treat this simply as a set, *i.e.* a discrete predomain.

Then we can define a domain-theoretic quantum monad:

$$T_{\mathcal{H}}(A) = \mathbf{D}(\mathcal{H}) \Rightarrow \mathcal{P}_{\mathsf{prob}}(A \times \mathbf{D}(\mathcal{H}))$$

Here $\mathcal{P}_{\mathsf{prob}}(\cdot)$ is the probabilistic powerdomain.

Further Directions

- Types
- Program logics
- Quantum concurrent processes. (This would (probably/possibly) involve the combined prob/non-det powerdomain).

See also: *A categorical semantics for quantum protocols*, S.A. and Bob Coecke, to appear in LiCS 2004. Available at www.arXiv.org/quant-ph/0402130.