Classical Control in Quantum Programs

Towards a quantum programming language

Dominique Unruh
IAKS
Universität Karlsruhe
unruh@iaks.uka.de

Operational Semantics

Programs may:

- Terminate or run forever
- Have classical output (even when not terminating)
- Have a post-execution state (when terminating)
- Realize any operation compatible with quantum mechanics

Programs modelled as:

- Measurement operators
- Mixture of POVM (for non-terminating) and generalized measurement (for terminating programs)
- Classical output is measurement outcome

Simple Control Structures

Simple control operations:

- Measurement operator easy to define
- `P;Q`: Sequential composition of programs
- `print`: Classical output
- `if/switch`: Case distinction (branching) based on measurement results

Examples:

`print a; print b` `print ab`	Equivalent programs outputting **ab**
`if (M) print 1`	If measurement **M** yields true, print **1**

`switch (M as m)`
 `print m`
 `print M` (shorthand)
	Outputs result of measuring **M**.

Loops

Loops:

- Harder to formalise
- No natural lattice structure → Fixpoint approach fails
- No suitable topology → Limit approach fails
- **Axiomatic approach** works: State some required properties of loops and show that these define loops uniquely

Examples:

`while (M) P`	Run program **P** while measurement **M** yields **true**.
`while (M)` `print N`	Measure **M**. If nonzero, measure **N**, output the outcome, and redo from start
`while (true)` `print x`	Outputs infinite sequence $\mathbf{x}^\infty$

Reasoning

Reasoning:

- Pre-/postconditions as sets of density operators
- Express equality of programs conditioned on initial states
- Reason about programs conditioned on some classical output

Examples:

{1} **P** {1}	If the initial state is a random state, so is the post-execution state (e.g. **P** is a permutation of basis states)
{**x** in computational basis} **P**=**noop**	If variable **x** is in the computational basis, program **P** has no effect (e.g. **P** might be a dephasing of **x**)
$\{\mathrm{tr}\rho = 1\}$ $\mathbf{P}\vert_{\mathbf{a}}$ $\{\mathrm{tr}\rho = ½\}$	Program **P** has probability ½ of outputting **a**

Example

H	`i:=0; while (i<n) { H₂ x[i]; i:=i+1 }`
U_f	`y:=0; y ¬ y ⊕ f(x); σ_y y; y ¬ y ⊕ f(x)`
U_0	`y:=0; y ¬ y ⊕ OR(x); σ_y y; y ¬ y ⊕ OR(x)`
Grover	see below

```
i:=0; while (i<n) { x[i]:=0; i:=i+1 };
H;
while (not f(x)) {
   while (k<r(n))
      U_f; H; U_0; H;
      k := k+1
   }};
i:=0; while (i<n) { print x[i]; i=i+1 }
```