Qtpi: Simulating (Concurrent) Quantum Systems

Richard Bornat

(Emeritus) Department of Computer Science, Middlesex University, London, UK

BCS FACS, 19th October 2021

To begin

Communicating Quantum Processes (Gay and Nagarajan, POPL 2005) describes a programming language.

To begin

Communicating Quantum Processes (Gay and Nagarajan, POPL 2005) describes a programming language.

An Introduction to Quantum Computing for Non-Physicists (Rieffel and Polak, ACM Computing Surveys, 2000) explains quantum calculation.

To begin

Communicating Quantum Processes (Gay and Nagarajan, POPL 2005) describes a programming language.

An Introduction to Quantum Computing for Non-Physicists (Rieffel and Polak, ACM Computing Surveys, 2000) explains quantum calculation.

Qtpi is an implementation of modified CQP, with a symbolic quantum calculator.

To begin

Communicating Quantum Processes (Gay and Nagarajan, POPL 2005) describes a programming language.

An Introduction to Quantum Computing for Non-Physicists (Rieffel and Polak, ACM Computing Surveys, 2000) explains quantum calculation.

Qtpi is an implementation of modified CQP, with a symbolic quantum calculator.

It's available on github at mdxtoc/qtpi, with lots of examples and documentation. Can do BB84 QKD and other protocols, Grover's algorithm, W state calculation, and more.

What's interesting?

1. ~~Quantum stuff~~
2. Language
3. Symbolic calculator
4. Probabilistic execution
5. Resource accounting for qubits
6. ~~Qbit collections~~
7. ~~Overloaded operators~~
8. ~~Sparse matrix tricks~~
9. ~~Iterative constructs~~
10. Demos

Protocol steps

A protocol agent can:

Protocol steps

A protocol agent can:

- obtain a qubit;

Protocol steps

A protocol agent can:

- obtain a qubit;
- put a qubit (or qubits) through a quantum gate;

Protocol steps

A protocol agent can:

- obtain a qubit;
- put a qubit (or qubits) through a quantum gate;
- measure a qubit;

Protocol steps

A protocol agent can:

- obtain a qubit;
- put a qubit (or qubits) through a quantum gate;
- measure a qubit;
- send or receive a qubit;

Protocol steps

A protocol agent can:

- obtain a qubit;
- put a qubit (or qubits) through a quantum gate;
- measure a qubit;
- send or receive a qubit;
- send or receive a classical value, such as a list of numbers or bits;

Protocol steps

A protocol agent can:

- obtain a qubit;
- put a qubit (or qubits) through a quantum gate;
- measure a qubit;
- send or receive a qubit;
- send or receive a classical value, such as a list of numbers or bits;
- do some classical calculation.

Process notation

Based on Milner's pi calculus. Very stark.

$$\textit{procdef} \quad ::= \quad p(x,\ \ldots, z) = P$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots, z) = P \\
P & ::= & \mathit{IO}\ .\ P \;\Big|\; \mathit{qstep}\ .\ P \;\Big|\; (\mathit{binder})\ .\ P
\end{array}
$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots,z) = P \\
P & ::= & \mathit{IO}\ .\ P \mid \mathit{qstep}\ .\ P \mid (\mathit{binder})\ .\ P \\
 & & \mid p(E,\ \ldots,E) \mid \mathit{par} \mid \mathit{alt} \mid \mathit{cond} \mid \_0
\end{array}
$$

Process notation

Based on Milner's pi calculus. Very stark.

$$\begin{array}{rcl} \mathit{procdef} & ::= & p(x, \ldots, z) = P \\ P & ::= & IO \,.\, P \mid \mathit{qstep} \,.\, P \mid (\mathit{binder}) \,.\, P \\ & & \mid p(E, \ldots, E) \mid \mathit{par} \mid \mathit{alt} \mid \mathit{cond} \mid \_0 \\ IO & ::= & C \,!\, E, \ldots, E \mid C \,?\, (x, \ldots, z) \end{array}$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x, \ldots, z) = P \\
P & ::= & \mathit{IO} \,.\, P \;\big|\; \mathit{qstep} \,.\, P \;\big|\; (\mathit{binder}) \,.\, P \\
& & \;\big|\; p(E, \ldots, E) \;\big|\; \mathit{par} \;\big|\; \mathit{alt} \;\big|\; \mathit{cond} \;\big|\; \_0 \\
\mathit{IO} & ::= & C \;!\; E, \ldots, E \;\big|\; C \;?\; (x, \ldots, z) \\
\mathit{qstep} & ::= & Q, \ldots, Q \gg G \;\big|\; q \not\leftarrow (x)
\end{array}
$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots,z) = P \\
P & ::= & \mathit{IO}\ .\ P \ \Big|\ \mathit{qstep}\ .\ P \ \Big|\ (\mathit{binder})\ .\ P \\
 & & \ \Big|\ p(E,\ \ldots,E) \ \Big|\ \mathit{par} \ \Big|\ \mathit{alt} \ \Big|\ \mathit{cond} \ \Big|\ \_0 \\
\mathit{IO} & ::= & C\ \texttt{!}\ E,\ \ldots,E \ \Big|\ C\ \texttt{?}\ (x,\ \ldots,z) \\
\mathit{qstep} & ::= & Q,\ \ldots,Q \gg G \ \Big|\ q \not\frown (x) \\
\mathit{binder} & ::= & \texttt{new}\ c \ \Big|\ \texttt{newq}\ q \ \Big|\ \texttt{newq}\ q = E \ \Big|\ \texttt{let}\ \mathit{pat} = E
\end{array}
$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots,z) = P \\
P & ::= & IO\ .\ P \ \big|\ \mathit{qstep}\ .\ P \ \big|\ (\mathit{binder})\ .\ P \\
 & & \big|\ p(E,\ \ldots,E) \ \big|\ \mathit{par} \ \big|\ \mathit{alt} \ \big|\ \mathit{cond} \ \big|\ \_0 \\
IO & ::= & C\ \texttt{!}\ E,\ \ldots,E \ \big|\ C\ \texttt{?}\ (x,\ \ldots,z) \\
\mathit{qstep} & ::= & Q,\ \ldots,Q \gg G \ \big|\ q \not\leftarrow (x) \\
\mathit{binder} & ::= & \texttt{new}\ c \ \big|\ \texttt{newq}\ q \ \big|\ \texttt{newq}\ q = E \ \big|\ \texttt{let}\ \mathit{pat} = E \\
\mathit{par} & ::= & [\ \mid\ ]\ P \mid\ \ldots\ \mid P \\
\mathit{alt} & ::= & [\ +\ ]\ IO\ .\ P +\ \ldots\ + IO\ .\ P \\
\mathit{cond} & ::= & \texttt{if}\ E\ \texttt{then}\ P\ \texttt{else}\ P \ \big|\ \texttt{match}\ E\ .\ \mathit{pat}.P +\ \ldots\ + \mathit{pat}.P
\end{array}
$$

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots,z) = P \\
P & ::= & IO\ .\ P \mid \mathit{qstep}\ .\ P \mid (\mathit{binder})\ .\ P \\
 & & \mid p(E,\ \ldots,E) \mid \mathit{par} \mid \mathit{alt} \mid \mathit{cond} \mid \_0 \\
IO & ::= & C\ \texttt{!}\ E,\ \ldots,E \mid C\ \texttt{?}\ (x,\ \ldots,z) \\
\mathit{qstep} & ::= & Q,\ \ldots,Q \gg G \mid q \not\frown (x) \\
\mathit{binder} & ::= & \texttt{new}\ c \mid \texttt{newq}\ q \mid \texttt{newq}\ q = E \mid \texttt{let}\ \mathit{pat} = E \\
\mathit{par} & ::= & [\ \texttt{|}\ ]\ P\ \texttt{|}\ \ldots\ \texttt{|}\ P \\
\mathit{alt} & ::= & [\ +\ ]\ IO\ .\ P\ +\ \ldots\ +\ IO\ .\ P \\
\mathit{cond} & ::= & \texttt{if}\ E\ \texttt{then}\ P\ \texttt{else}\ P \mid \texttt{match}\ E\ .\ \mathit{pat}.P\ +\ \ldots\ +\ \mathit{pat}.P
\end{array}
$$

Communication channels can be sent and received in messages

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x,\ \ldots,z) = P \\
P & ::= & IO\ .\ P \;\big|\; \mathit{qstep}\ .\ P \;\big|\; (\mathit{binder})\ .\ P \\
 & & \big|\; p(E,\ \ldots,E) \;\big|\; \mathit{par} \;\big|\; \mathit{alt} \;\big|\; \mathit{cond} \;\big|\; \_0 \\
IO & ::= & C\ \texttt{!}\ E,\ \ldots,E \;\big|\; C\ \texttt{?}\ (x,\ \ldots,z) \\
\mathit{qstep} & ::= & Q,\ \ldots,Q \gg G \;\big|\; q \not\downarrow (x) \\
\mathit{binder} & ::= & \texttt{new}\ c \;\big|\; \texttt{newq}\ q \;\big|\; \texttt{newq}\ q = E \;\big|\; \texttt{let}\ \mathit{pat} = E \\
\mathit{par} & ::= & [\ \texttt{|}\ ]\ P\ \texttt{|}\ \ldots\ \texttt{|}\ P \\
\mathit{alt} & ::= & [\ \texttt{+}\ ]\ IO\ .\ P\ \texttt{+}\ \ldots\ \texttt{+}\ IO\ .\ P \\
\mathit{cond} & ::= & \texttt{if}\ E\ \texttt{then}\ P\ \texttt{else}\ P \;\big|\; \texttt{match}\ E\ .\ \mathit{pat}.P\ \texttt{+}\ \ldots\ \texttt{+}\ \mathit{pat}.P
\end{array}
$$

Communication channels can be sent and received in messages; so can qubits.

Process notation

Based on Milner's pi calculus. Very stark.

$$
\begin{array}{rcl}
\mathit{procdef} & ::= & p(x, \ldots, z) = P \\
P & ::= & IO \;.\; P \;\big|\; \mathit{qstep} \;.\; P \;\big|\; (\mathit{binder}) \;.\; P \\
& & \big|\; p(E, \ldots, E) \;\big|\; \mathit{par} \;\big|\; \mathit{alt} \;\big|\; \mathit{cond} \;\big|\; \_0 \\
IO & ::= & C \;\texttt{!}\; E, \ldots, E \;\big|\; C \;\texttt{?}\; (x, \ldots, z) \\
\mathit{qstep} & ::= & Q, \ldots, Q \gg G \;\big|\; q \not\leftarrow (x) \\
\mathit{binder} & ::= & \texttt{new}\; c \;\big|\; \texttt{newq}\; q \;\big|\; \texttt{newq}\; q = E \;\big|\; \texttt{let}\; \mathit{pat} = E \\
\mathit{par} & ::= & [\;|\;]\; P \;|\; \ldots \;|\; P \\
\mathit{alt} & ::= & [\;+\;]\; IO \;.\; P + \ldots + IO \;.\; P \\
\mathit{cond} & ::= & \texttt{if}\; E \;\texttt{then}\; P \;\texttt{else}\; P \;\big|\; \texttt{match}\; E \;.\; \mathit{pat}.P + \ldots + \mathit{pat}.P
\end{array}
$$

Communication channels can be sent and received in messages; so can qubits.

Qtpi is strongly typed (Hindley-Milner). Explicit typing is optional (but, for simplicity, not described).

Expression notation

There's a functional notation for calculation, which looks like Miranda ...

Expression notation

There's a functional notation for calculation, which looks like Miranda ... but it's eager.

Expression notation

There's a functional notation for calculation, which looks like Miranda ... but it's eager.

To deal with qubit accounting, functions can't have anything to do with qubits.

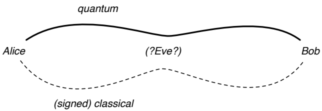
What's interesting (again)?

1. ~~Quantum stuff~~
2. ~~Language~~
3. Symbolic calculator
4. Probabilistic execution
5. Resource accounting for qubits
6. ~~Qbit collections~~
7. ~~Overloaded operators~~
8. ~~Sparse matrix tricks~~
9. ~~Iterative constructs~~
10. Demos

Demo

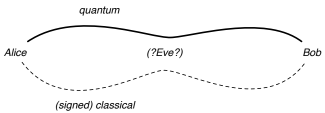
Teleportation, queen of the baby protocols

BB84, queen of the QKD protocols



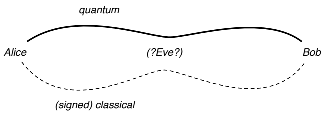
- generate a one-time code without transmitting it.

BB84, queen of the QKD protocols



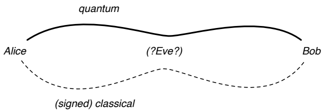
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);

BB84, queen of the QKD protocols



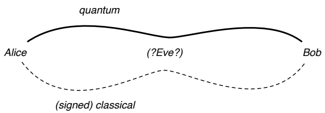
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;

BB84, queen of the QKD protocols



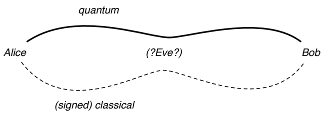
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);

BB84, queen of the QKD protocols



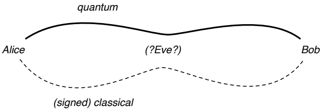
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;

BB84, queen of the QKD protocols



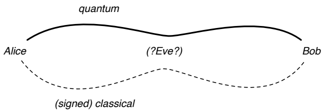
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;
- If Eve has not intervened, they share $\sim$500 secret bits;

BB84, queen of the QKD protocols



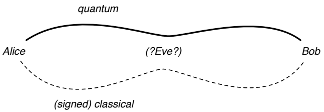
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;
- If Eve has not intervened, they share $\sim$500 secret bits;
- Bob sends Alice (classically) a random sample of n of his bits;

BB84, queen of the QKD protocols



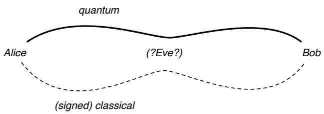
- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;
- If Eve has not intervened, they share $\sim$500 secret bits;
- Bob sends Alice (classically) a random sample of n of his bits;
- Only a $(\frac{3}{4})^n$ chance that Eve has meddled and those bits match;

BB84, queen of the QKD protocols



- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;
- If Eve has not intervened, they share $\sim$500 secret bits;
- Bob sends Alice (classically) a random sample of n of his bits;
- Only a $(\frac{3}{4})^n$ chance that Eve has meddled and those bits match;
- Otherwise A & B share a (500-n)-bit secret one-time code;

BB84, queen of the QKD protocols



- generate a one-time code without transmitting it.
- Alice chooses 1000 bits (say);
- Alice sends them as 1000 qubits, randomly choosing diagonal ($|+\rangle$, $|-\rangle$) or normal ($|0\rangle$, $|1\rangle$) encoding for 0 and 1;
- Bob measures them, randomly as diagonal or normal (50/50 he guesses right on each);
- They compare notes (classically) about their random choices;
- If Eve has not intervened, they share $\sim$500 secret bits;
- Bob sends Alice (classically) a random sample of n of his bits;
- Only a $(\frac{3}{4})^n$ chance that Eve has meddled and those bits match;
- Otherwise A & B share a (500-n)-bit secret one-time code;
- Alice uses it to XOR the message and send it classically.