Some Recent Research Issues in Quantum Logic

Marek Perkowski

Part one
What will be discussed?

- 1. Background
- 2. Quantum circuits synthesis
- 3. Quantum circuits simulation
- 4. Quantum logic emulation and evolvable hardware
- 5. Quantum circuits verification
- 6. Quantum-based robot control
Quantum Logic Circuits

Origin of slides: John Hayes, Peter Shor, Martin Lukac, Mikhail Pivtoraiko, Alan Mishchenko, Pawel Kerntopf.
The simplest explanation is that the beam-splitter acts as a classical coin-flip, randomly sending each photon one way or the other.
The simplest explanation must be wrong, since it would predict a 50-50 distribution.
More experimental data

\[ \sin^2\left(\frac{\phi}{2}\right) \]

\[ \cos^2\left(\frac{\phi}{2}\right) \]
The particle can exist in a linear combination or superposition of the two paths.

\[ \frac{i}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \]

\[ \frac{i}{\sqrt{2}} |0\rangle + \frac{e^{i\phi}}{\sqrt{2}} |1\rangle \]

\[ \frac{e^{i\phi} - 1}{2} |0\rangle + \frac{i(e^{i\phi} + 1)}{\sqrt{2}} |1\rangle \]
If the photon is measured when it is in the state $\alpha_0 |0\rangle + \alpha_1 |1\rangle$ then we get $|0\rangle$ with probability $|\alpha_0|^2$

\[ |\alpha_0|^2 + |\alpha_1|^2 = 1 \]
Quantum Operations

The operations are induced by the apparatus *linearly*, that is, if

\[
|0\rangle \rightarrow \frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle
\]

and

\[
|1\rangle \rightarrow \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle
\]

then

\[
\alpha_0|0\rangle + \alpha_1|1\rangle \rightarrow \alpha_0\left(\frac{i}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) + \alpha_1\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle\right)
\]

\[
= \left(\alpha_0 \frac{i}{\sqrt{2}} + \alpha_1 \frac{1}{\sqrt{2}}\right)|0\rangle + \left(\alpha_0 \frac{1}{\sqrt{2}} + \alpha_1 \frac{i}{\sqrt{2}}\right)|1\rangle
\]
Quantum Operations

Any linear operation that takes states
\[ \alpha_0 |0\rangle + \alpha_1 |1\rangle \] satisfying
\[ |\alpha_0|^2 + |\alpha_1|^2 = 1 \]
and maps them to states
\[ \alpha'_0 |0\rangle + \alpha'_1 |1\rangle \] satisfying
\[ |\alpha'_0|^2 + |\alpha'_1|^2 = 1 \]
must be UNITARY
$\left| 0 \right\rangle$ corresponds to \begin{pmatrix} 1 \\ 0 \end{pmatrix}

$\left| 1 \right\rangle$ corresponds to \begin{pmatrix} 0 \\ 1 \end{pmatrix}

$\alpha_0 \left| 0 \right\rangle + \alpha_1 \left| 1 \right\rangle$ corresponds to \[ \alpha_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \alpha_1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} \]
corresponds to \[
\begin{pmatrix}
i & 1 \\
\sqrt{2} & \sqrt{2} \\
1 & i \\
\sqrt{2} & \sqrt{2}
\end{pmatrix}
\]
corresponds to \[
\begin{pmatrix}
1 & 0 \\
0 & e^{i\varphi}
\end{pmatrix}
\]
corresponds to

$$\begin{pmatrix}
\frac{i}{\sqrt{2}} & 1 \\
1 & \frac{i}{\sqrt{2}}
\end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix} \begin{pmatrix}
\frac{i}{\sqrt{2}} & 1 \\
\frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}}
\end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$
Linear Algebra

\[ U = \begin{bmatrix}
    u_{00} & u_{01} \\
    u_{10} & u_{11}
\end{bmatrix} \]

is unitary if and only if

\[ UU^* = \begin{bmatrix}
    u_{00} & u_{01} \\
    u_{10} & u_{11}
\end{bmatrix}\begin{bmatrix}
    u_{00}^* & u_{10}^* \\
    u_{01}^* & u_{11}^*
\end{bmatrix} = \begin{bmatrix}
    1 & 0 \\
    0 & 1
\end{bmatrix} = I \]
The two position states of a photon in a Mach-Zehnder apparatus is just one example of a quantum bit or qubit

Except when addressing a particular physical implementation, we will simply talk about “basis” states $|0\rangle$ and $|1\rangle$ and unitary operations like

$$
\begin{align*}
\text{H} & \quad \text{and} \quad \phi
\end{align*}
$$
where $H$ corresponds to \[
\begin{pmatrix}
\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\
1 & -1 \\
\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}
\end{pmatrix}
\]

and $\varphi$ corresponds to \[
\begin{pmatrix}
1 & 0 \\
0 & e^{i\varphi}
\end{pmatrix}
\]
An arrangement like

\[ 0 \]

is represented with a network like

\[ |0\rangle \rightarrow H \rightarrow \varphi \rightarrow H \]
More than one qubit

If we concatenate two qubits

\[(\alpha_0 |0\rangle + \alpha_1 |1\rangle) (\beta_0 |0\rangle + \beta_1 |1\rangle)\]

we have a 2-qubit system with 4 basis states

\[|00\rangle = |0\rangle |0\rangle, \quad |01\rangle = |0\rangle |1\rangle, \quad |10\rangle = |1\rangle |0\rangle, \quad |11\rangle = |1\rangle |1\rangle\]

and we can also describe the state as

\[\alpha_0 \beta_0 |00\rangle + \alpha_0 \beta_1 |01\rangle + \alpha_1 \beta_0 |10\rangle + \alpha_1 \beta_1 |11\rangle\]

or by the vector

\[
\begin{pmatrix}
\alpha_0 \beta_0 \\
\alpha_0 \beta \\
\alpha_1 \beta_0 \\
\alpha_1 \beta_1 \\
\end{pmatrix} =
\begin{pmatrix}
\alpha_0 \\
\alpha_1 \\
\end{pmatrix} \otimes
\begin{pmatrix}
\beta_0 \\
\beta_1 \\
\end{pmatrix}
\]
More than one qubit

In general we can have arbitrary superpositions

\[ \alpha_{00}|0\rangle|0\rangle + \alpha_{01}|0\rangle|1\rangle + \alpha_{10}|1\rangle|0\rangle + \alpha_{11}|1\rangle|1\rangle \]

\[ |\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1 \]

where there is no factorization into the tensor product of two independent qubits. These states are called entangled.
Measuring multi-qubit systems

If we measure both bits of
\[ \alpha_{00}|0\rangle|0\rangle + \alpha_{01}|0\rangle|1\rangle + \alpha_{10}|1\rangle|0\rangle + \alpha_{11}|1\rangle|1\rangle \]

we get \(|\times\rangle|\gamma\rangle\) with probability \(|\alpha_{xy}|^2\)
Classical

Versus

Quantum
Goal: Fast, low-cost implementation of useful algorithms using standard components (gates) and design techniques

Classical Logic Circuits
- Circuit behavior is governed implicitly by classical physics
- Signal states are simple bit vectors, e.g. $X = 01010111$
- Operations are defined by Boolean Algebra
- No restrictions exist on copying or measuring signals
- Small well-defined sets of universal gate types, e.g. \{NAND\}, \{AND,OR,NOT\}, \{AND,NOT\}, etc.
- Well developed CAD methodologies exist
- Circuits are easily implemented in fast, scalable and macroscopic technologies such as CMOS
• **Quantum Logic Circuits**
  – Circuit behavior is governed explicitly by quantum mechanics
  – Signal states are vectors interpreted as a **superposition** of binary “qubit” vectors with complex-number coefficients

\[
|\Psi\rangle = \sum_{i=0}^{2^n-1} c_i |i_{n-1}i_{n-1}...i_0\rangle
\]

  – Operations are defined by linear algebra over Hilbert Space and can be represented by unitary matrices with complex elements
  – **Severe restrictions** exist on copying and measuring signals
  – Many universal gate sets exist but the best types are not obvious
  – Circuits must use microscopic technologies that are slow, fragile, and not yet scalable, e.g., NMR
Unitary Operations

- Gates and circuits must be reversible (information-lossless)
  - Number of output signal lines = Number of input signal lines
  - The circuit function must be a bijection, implying that output vectors are a permutation of the input vectors
- Classical logic behavior can be represented by permutation matrices
- Non-classical logic behavior can be represented including state sign (phase) and entanglement
Quantum Circuit Characteristics

- Quantum Measurement
  - Measurement yields only one state $X$ of the superposed states
  - Measurement also makes $X$ the new state and so interferes with computational processes
  - $X$ is determined with some probability, implying uncertainty in the result
  - States cannot be copied (“cloned”), implying that signal fanout is not permitted
  - Environmental interference can cause a measurement-like state collapse (decoherence)
Classical vs. Quantum Circuits

Classical adder

```
\begin{align*}
  c_n &= a_n + b_n + c_{n-1} \\
  s_0 &= a_0 \cdot b_0 \\
  s_1 &= a_0 \cdot b_1 + a_1 \cdot b_0 \\
  s_2 &= a_1 \cdot b_1 + a_0 \cdot b_2 + a_1 \cdot b_0 + a_2 \cdot b_0 \\
  s_3 &= a_1 \cdot b_1 + a_0 \cdot b_2 + a_1 \cdot b_0 + a_2 \cdot b_0 + a_3 \cdot b_0 \\
  c_n &= a_n \cdot b_n + a_n \cdot c_{n-1} + b_n \cdot c_{n-1}
\end{align*}
```
Classical vs. Quantum Circuits

Quantum adder

\[
\begin{align*}
    c_{n-1} &= 0 \\
    a_{n-1} \\
    b_{n-1} \\
    c_n &= 0 \\
    a_n \\
    b_n \\
    b_{n+1} &= 0 \\
    c_{n-1} &= 0 \\
    a_{n-1} \\
    b_{n-1}' \\
    c_n &= 0 \\
    a_n \\
    b_n' \\
    b_{n+1}' \\
\end{align*}
\]
Reversible Circuits
Reversible Circuits

- Reversibility was studied around 1980 motivated by power minimization considerations.
- Bennett, Toffoli et al. showed that any classical logic circuit $C$ can be made reversible with modest overhead.
Reversible Circuits

- How to make a given $f$ reversible
  - Suppose $f : i \rightarrow f(i)$ has $n$ inputs $m$ outputs
  - Introduce $n$ extra outputs and $m$ extra inputs
  - Replace $f$ by $f_{\text{rev}} : i, j \rightarrow i, f(i) \oplus j$ where $\oplus$ is XOR

- Example 1: $f(a, b) = \text{AND}(a, b)$

- This is the well-known Toffoli gate, which realizes AND when $c = 0$, and NAND when $c = 1$. 

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>a</th>
<th>b</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Reversible Circuits

- Reversible gate family [Toffoli 1980]

- Every Boolean function has a reversible implementation using Toffoli gates.

- There is no universal reversible gate with fewer than three inputs
Quantum
Gates
Quantum Gates

- **One-Input gate: NOT**
  - Input state: $c_0|0\rangle + c_1|1\rangle$
  - Output state: $c_1|0\rangle + c_0|1\rangle$
  - Pure states are mapped thus: $|0\rangle \rightarrow |1\rangle$ and $|1\rangle \rightarrow |0\rangle$
  - Gate operator (matrix) is
    $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
    $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
    $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
  - As expected:
    $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
Quantum Gates

- **One-Input gate:** “Square root of NOT”
  - Some matrix elements are imaginary
  - Gate operator (matrix):
    \[
    \begin{pmatrix}
    i/\sqrt{1/2} & 1/\sqrt{1/2} \\
    1/\sqrt{1/2} & i/\sqrt{1/2}
    \end{pmatrix}
    = \frac{1}{\sqrt{2}} \begin{pmatrix} i & 1 \\ 1 & i \end{pmatrix}
    \]
  - We find:
    \[
    \frac{1}{\sqrt{2}} \begin{pmatrix} i & 1 \\ 1 & i \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} i \\ 1 \end{pmatrix}
    \]
    so \( |0\rangle \rightarrow |0\rangle \) with probability \( |i/\sqrt{2}|^2 = 1/2 \)
    \[
    \frac{1}{\sqrt{2}} \begin{pmatrix} i & 1 \\ 1 & i \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} i \\ 1 \end{pmatrix}
    \]
    and \( |0\rangle \rightarrow |1\rangle \) with probability \( |1/\sqrt{2}|^2 = 1/2 \)
    
    Similarly, this gate randomizes input \(|1\rangle\)
  - But concatenation of two gates eliminates the randomness!

\[
\frac{1}{2} \begin{pmatrix} i & 1 \\ 1 & i \end{pmatrix} \begin{pmatrix} i & 1 \\ i & 1 \end{pmatrix} = \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix}
\]
Quantum Gates

• One-Input gate: Hadamard

\[
\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}
\]

– Maps \( |0\rangle \rightarrow \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \) and \( |1\rangle \rightarrow \frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle \).

– Ignoring the normalization factor \( \frac{1}{\sqrt{2}} \), we can write \( |x\rangle \rightarrow (-1)^x |x\rangle - |1 - x\rangle \).

• One-Input gate: Phase shift

\[
\begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}
\]
Universal One-Input Gate Sets

- Requirement:

\[ |0\rangle \quad \xrightarrow{U} \quad \text{Any state} \quad |\psi\rangle \]

- **Hadamard** and **phase-shift** gates form a **universal** gate set

- **Example:** The following circuit generates

\[ |\psi\rangle = \cos \theta \ |0\rangle + e^{i\phi} \sin \theta \ |1\rangle \] up to a global factor
Quantum Gates

- **Two-Input Gate: Controlled NOT (CNOT)**

\[ |x\rangle \xrightarrow{\text{CNOT}} |x\rangle \]

\[ |y\rangle \xrightarrow{\text{CNOT}} |x \oplus y\rangle \]

\[ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \]

- CNOT maps \(|x\rangle|0\rangle \rightarrow |x\rangle||x\rangle\) and \(|x\rangle|1\rangle \rightarrow |x\rangle||\text{NOT} \ x\rangle\)

\(|x\rangle|0\rangle \rightarrow |x\rangle||x\rangle\) *looks like cloning*, but it’s not. These mappings are **valid only for the pure states** \(|0\rangle\) and \(|1\rangle\)

- Serves as a “non-demolition” measurement gate
**Quantum Gates**

- **3-Input gate:** Controlled CNOT (C^2NOT or Toffoli gate)

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
\]

- Diagram:

\[
|a\rangle \quad |a\rangle \\
\\
|b\rangle \quad |b\rangle \\
\\
|c\rangle \quad |ab \oplus c\rangle
\]
Quantum Gates

- General controlled gates that control some 1-qubit unitary operation $U$ are useful

$$\begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$$

etc.
Universal Gate Sets

- To implement any unitary operation on $n$ qubits exactly requires an infinite number of gate types.
- The (infinite) set of all 2-input gates is universal.
  - Any $n$-qubit unitary operation can be implemented using $\Theta(n^{34^n})$ gates [Reck et al. 1994].
- CNOT and the (infinite) set of all 1-qubit gates is universal.
Discrete Universal Gate Sets

• The **error** on implementing $U$ by $V$ is defined as

$$E(U, V) = \max_{|\Psi\rangle} \| (U - V)|\Psi\rangle \|$$

• If $U$ can be implemented by $K$ gates, we can simulate $U$ with a total error less than $\varepsilon$ with a gate overhead that is polynomial in $\log(K/\varepsilon)$

• A discrete set of gate types $G$ is **universal**, if we can approximate any $U$ to within any $\varepsilon > 0$ using a sequence of gates from $G$
Quantum Gates

Discrete Universal Gate Set

- **Example 1**: Four-member “standard” gate set
  
  \[
  \begin{pmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 0 & 1 \\
  0 & 0 & 1 & 0
  \end{pmatrix}
  \]

  \[
  \frac{1}{\sqrt{2}} \begin{pmatrix}
  1 & 1 \\
  1 & -1
  \end{pmatrix}
  \]

  \[
  \begin{pmatrix}
  1 & 0 \\
  0 & i
  \end{pmatrix}
  \]

  \[
  \begin{pmatrix}
  1 & 0 \\
  0 & e^{i\pi/4}
  \end{pmatrix}
  \]

  CNOT          Hadamard      Phase      $\pi/8$ (T) gate

- **Example 2**: \{CNOT, Hadamard, Phase, Toffoli\}
Quantum Circuits
A quantum (combinational) circuit is a sequence of quantum gates, linked by “wires”

The circuit has fixed “width” corresponding to the number of qubits being processed

Logic design (classical and quantum) attempts to find circuit structures for needed operations that are
  – Functionally correct
  – Independent of physical technology
  – Low-cost, e.g., use the minimum number of qubits or gates

Quantum logic design is not well developed!
Ad hoc designs known for many specific functions and gates

**Example 1** illustrating a theorem by [Barenco et al. 1995]: Any $C^2(U)$ gate can be built from CNOTs, $C(V)$, and $C(V^\dagger)$ gates, where $V^2 = U$

\[ \begin{align*}
V^2 &= U \\
V &\downarrow \\
V^\dagger &\downarrow \\
V &\downarrow
\end{align*} \]
Example 1: Simulation

\[ |0\rangle |0\rangle = |0\rangle |0\rangle |0\rangle |0\rangle |0\rangle |0\rangle |0\rangle \]

\[ |1\rangle |1\rangle \]

\[ |x\rangle U |x\rangle \]

\[ |x\rangle V |x\rangle \]

\[ V |x\rangle \]

\[ V^\dagger |x\rangle \]

\[ V |x\rangle \]
Example 1: Simulation (contd.)

- Example: Simulate the two remaining cases

- Exercise: Simulate the two remaining cases
Example 1: Algebraic analysis

\[ U_4 U_2 U_3 U_1(x_1, x_2, x_3) \]

Is

\[ U_0(x_1, x_2, x_3) = U_5 U_4 U_3 U_2 U_1(x_1, x_2, x_3) \]

\[ = (x_1, x_2, x_1 x_2 \oplus U(x_3)) \] ?
Example 1 (contd);

\[ U_1 = I_1 \otimes C(V) \]

\[
= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \nu_{00} & \nu_{01} \\ 0 & 0 & \nu_{10} & \nu_{11} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \nu_{00} & \nu_{01} & 0 & 0 & 0 & 0 \\ 0 & 0 & \nu_{10} & \nu_{11} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \nu_{00} & \nu_{01} \\ 0 & 0 & 0 & 0 & 0 & 0 & \nu_{10} & \nu_{11} \end{pmatrix}
\]
Example 1 (contd);

\[ U_2 = U_4 = CNOT(x_1, x_2) \otimes I_1 \]

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}
\otimes
\begin{pmatrix}
1 & 0 \\
0 & 1
\end{pmatrix}
= 
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix}
\]
Quantum Circuits

Example 1 (contd);

- $U_5$ is the same as $U_1$ but has $x_1$ and $x_2$ permuted (tricky!)
- It remains to evaluate the product of five 8 x 8 matrices $U_5 U_4 U_3 U_2 U_1$ using the fact that $VV^\dagger = I$ and $VV = U$

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} = U_0
$$
Implementing a Half Adder

- Problem: Implement the classical functions $sum = x_1 \oplus x_0$ and $carry = x_1x_0$

Generic design:

$$\begin{align*}
|x_1\rangle & \quad U_{add} \quad |x_1\rangle \\
|x_0\rangle & \quad |x_0\rangle \\
|y_1\rangle & \quad |y_1\rangle \oplus carry \\
|y_0\rangle & \quad |y_0\rangle \oplus sum
\end{align*}$$
• **Half Adder.** Generic design (contd.)

\[
U_{ADD} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]
Quantum Circuits

- **Half Adder**: Specific (reduced) design

$$\begin{align*}
|x_1\rangle & \quad C^2\text{NOT} \quad \text{(Toffoli)} \\
|x_0\rangle & \quad \text{CNOT} \\
|y\rangle & \quad |x_1\rangle \quad \text{sum} \\
& \quad |y\rangle \oplus \text{carry}
\end{align*}$$
Walsh Transform for two binary-input many-valued variables

Classical logic

Quantum logic

You need a butterfly

Butterfly is created automatically by tensor product corresponding to superposition
Portland Quantum Logic Group (PQLG)

What we do?
People at PSU and collaborators

- Marek Perkowski
- Martin Zwick
- Xiaoyu Song
- William Hung
- Anas Al-Rabadi
- Martin Lukac
- Mikhail Pivtoraiko
- Andrei Khlopotine
- Alan Mishchenko (University of California, Berkeley, USA)
- Bernd Steinbach (Technical University of Freiberg, Germany)
- Pawel Kerntopf (Technical University of Warsaw, Poland)
- Mitch Thornton (Southern Methodist University, Dallas, USA)
- Lech Jozwiak (Technical University of Eindhoven, The Netherlands)
- Andrzej Buller (ATR, Kansai Science City, Japan)
- Tsutomu Sasao (Kyushu University of Technology, Iizuka, Japan).
Current Projects

• Logic Synthesis for Reversible Logic
  – decomposition
  – Decision Diagram Mapping
  – composition
  – regular structures - lattices, PLAs, nets

• Logic Synthesis for Quantum Logic

• Quantum Simulation using new Decision Diagrams

4 papers published

paper submitted
Current Projects

- FPGA-based model of Quantum Computer
- Reversible FPGA using CMOS.
- Realization of new spectral transforms using quantum logic.
- Non-linear Quantum Logic solves NP problems in polynomial time.
- Quantum-inspired search algorithms for robotics
Where to learn more

- Web Page of Marek Perkowski
  - class 572 - *see description of student projects*
  - Portland Quantum Logic Group

We are open to collaboration and we want to grow
Automated Synthesis of Generalized Reversible Cascades using Genetic Algorithm
Agenda

- Introduction and history
- Reversible Logic and Reversible Gates
- Genetic algorithms
- The Model
- Simulation
- Conclusion
Reversible gates...

Feynman, Toffoli, Fredkin, …

\[ A \rightarrow P = A \]
\[ B \rightarrow Q = P \oplus B \]

Mapping of I/O allows unique \((P,Q) \Rightarrow (A,B)\)

and Reversible Circuits

- To reduce the RL synthesis limitations one can insert constants in order to modify the functionality
Generalized Reversible Gates

n - inputs

A
B
P
Q
f
C
R

A
B
P
Q
f
C
R

A
B
P
Q
f
C
R

A
B
P
Q
f
C
R

A
B
P
Q
f
C
R

A
B
C
f

0
1

0
1
Perkowski gates family

Derivatives of Perkowski’s Gate

Generalized multi-input multiplexer

k

f2

t

g

h

Many other gate families

Kerntopf gates

Fredkin gates

De Vos gates

f2

De Vos gates

f

Feynman gates

CMOS gates

A

B

C

D

E

R

S

A

B

C

R

P

Q

P

Q

A

B

C

R

P

Q

A

B

C

R

P

Q

Generalized Maitra gates

Maitra gates
Cascades

- Mixed data/control inputs (generalized complex control gates)
- All:
  - ESOP
  - Factorized-ESOP
  - MV Complex Terms
  - XOR family

Example:
Genetic algorithms

- Population of n individuals
- Chromosome variable length
- Parallel blocks
- Classic GA’s operators
Encoding & operations

Mutation

C-O
Circuit Encoding

Toffoli

Feynmann

Inverter

Wire

Feynmann

Feynmann

Circuit LUT representation

4 /PWT/P /PFI/P /PFF/P
GA’s settings

- Stochastic universal sampling
- Fitness:
  \[ F_i = \frac{1}{1 + \text{error}_i} - \Lambda_i \]
- Error:
  \[ \text{error} = \sum_{i=1}^{n} \sum_{j=1}^{2^n} |U_{ij} - S_{ij}| \quad S, U \in U(2^n) \]
- Error evaluation:
  - Comparison outputs / LUT
  - Permutations of all constants and inputs
  - Normalization of error by wires and patterns
  - Penalization for length

LUT for Fredkin gate:

<table>
<thead>
<tr>
<th>A, B, C</th>
<th>A'</th>
<th>B'</th>
<th>C'</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Mutation</td>
<td>Gates Blocks</td>
<td>Position (block/circuit)</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------------</td>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td>Cross-Over*</td>
<td>Segments</td>
<td>Experimental (unitary matrices)</td>
<td></td>
</tr>
<tr>
<td>Reproduction</td>
<td>Circuits</td>
<td>Best gates Best Circuits</td>
<td></td>
</tr>
</tbody>
</table>

* - for circuits having only same number of I/O
Each input is equivalent with any other
Evolving new circuits by recombination
Non specific conditions
Population 100-150
Mutation = 0.01 –1
Crossover = 0.3 – 0.8
Specifications:
- Genetic operations based on RCB > minimal element
- The noise in these experiments is not only a mutation but an random operator on random blocks !!!
Testing

- No starting set restriction
- Mutation only on blocks

Unitary gate search

<table>
<thead>
<tr>
<th># of inputs</th>
<th>Number of individuals</th>
<th>Number of generations</th>
<th>Real gate found</th>
<th>Real Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10/50</td>
<td>10/1</td>
<td>*</td>
<td>&lt; 1 Min</td>
</tr>
<tr>
<td>3</td>
<td>10/50</td>
<td>10/1</td>
<td>*</td>
<td>&lt; 1 Min</td>
</tr>
<tr>
<td>4</td>
<td>10/50</td>
<td>10/1</td>
<td>*</td>
<td>&lt; 1 Min</td>
</tr>
</tbody>
</table>

Random function search
Improvements

- Using $\text{min}(\text{ESOP}(F\oplus G))$ for fitness
- Lamarckian learning
  - One genotype $\Rightarrow$ multiple possibilities of phenotype
  - Using to minimize Exorcism-4
Circuit search

- Starting set restriction
- Mutation all levels (0.01 – 0.1)

<table>
<thead>
<tr>
<th>Circuit/Gate</th>
<th># of Gen.</th>
<th>R.T.</th>
<th>Exact/similar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toffoli</td>
<td>5/1</td>
<td>0</td>
<td><em>/</em></td>
</tr>
<tr>
<td>Fredkin</td>
<td>5/1</td>
<td>0</td>
<td><em>/</em></td>
</tr>
<tr>
<td>Adder</td>
<td>?/200,000</td>
<td>120 sec</td>
<td>0/*</td>
</tr>
</tbody>
</table>
Conclusion

- **Ideas:**
  - Using GA to evolve arbitrary Reversible Circuit
  - Specific Encoding helps the evolution
  - Alternative encoding presented

- **Future works:**
  - Apply Lamarckian GA and other new variants of evolutionary approaches
  - Create hybrid algorithms by mixing evolutionary and logic-symbolic methods
  - Use new representations such as permutations and decision diagrams
  - Use Logic minimizer to minimize the ESOP expression of the circuit

- **THIS IS WORK IN PROGRESS, EVERYBODY IS WELCOME TO JOIN.**

- **Publishing**