Addition by Fourier transform

Carsten Urbach

This corresponds to problem 5.6 in Nielsen & Chuang. The original paper is (Draper 2000). Which quantum circuit can be used to perform the computation

$$|x\rangle \rightarrow |x + y \mod 2^n\rangle$$

with \(0 \leq x < 2^n\) and a constant integer \(y\).

We exploit the general idea

$$x + y = \log (e^x e^y)$$

where the exponentiation is de facto performed by a Fourier transform and the logarithm by the inverse transform.

Fourier transforming the state \(|x\rangle\) with \(n\) bits, leads to the following product representation

$$|x\rangle = |x_n x_{n-1} \ldots x_1\rangle \rightarrow \frac{1}{2^n}(|0\rangle + e^{2\pi i 0. x_1/2}|1\rangle)(|0\rangle + e^{2\pi i 0. x_2 x_1/2}|1\rangle) \cdots (|0\rangle + e^{2\pi i 0. x_n \ldots x_1/2}|1\rangle)$$

where we use the notation

$$x = x_1 2^0 + x_2 2^1 + \ldots + x_n 2^{n-1}$$

and

$$0. x_1 \ldots x_1 \equiv \frac{x_1}{2} + \frac{x_1-1}{2^2} + \ldots + \frac{x_1}{2^n}.$$

Now, we apply a phase shift \(R_\theta(\theta)\) to each qubit

$$R_z = \begin{pmatrix} 1 & 0 \\ 0 & \exp(i\theta) \end{pmatrix}.$$

We apply \(R_\theta\) with \(\theta_j = 2\pi y/2^{n-j-1}\) to qubit \(j\) where \(1 \leq j \leq n\). For \(y\) we can also write

$$y = y_1 2^0 + y_2 2^1 + \ldots + y_n 2^{n-1}.$$

Thus,

$$\exp(2\pi i y/2^{n-j+1}) = \prod_{k=0}^{n-1} \exp(2\pi i y_{k+1} 2^{j-1-n+k}).$$

Since \(\exp(2\pi iy_k l) = 1\) for positive integer \(l\), this reduces to (recall \(y_k \in \{0, 1\}\))

$$\exp(2\pi i y/2^{n-j+1}) = \prod_{k=0}^{n-j} \exp(2\pi i y_{k+1} 2^{j-1-n+k}).$$

The \(n\)th qubit gets multiplied with \(\exp(i\theta_n)\) with \(\theta_n = 2\pi y/2^1\). Thus, we need to compute

$$\exp(2\pi i x_1/2) \cdot \exp(2\pi i y_1/2) = \exp(2\pi i (x_1 + y_1)/2).$$

Similarly, for the \(j\)th qubit one gets

$$\exp(2\pi i (x_1/2^{n-j+1} + x_2/2^{n-j} + \ldots)) \cdot \exp(2\pi i (y_1/2^{n-j+1} + y_2/2^{n-j} + \ldots)) = \exp(2\pi i ((x_1+y_1)/2^{n-j+1}+(x_2+y_2)/2^{n-j}+\ldots)).$$
which implements the addition $\mod n$ operation in this binary fraction.

Now apply the inverse Fourier transform and it is easy to see that this transforms back to the state $|x + y \mod n\rangle$.

For the practical implementation we first need the phase shift operators, which is up to a phase identical to $R_z$:

\[
R_{\theta} <- \text{function}(\text{bit, theta=0.}) \{
\text{return}(\text{methods::new("sqgate", bit=as.integer(bit),}
\quad \text{M=as.complex(c(1, 0, 0, exp(1i*theta)),}
\quad \text{dim=c(2,2)), type="Rt"}))
\}
\]

With this one can write the desired function on state $x$.

\[
\text{addbyqft <- function(x, y)} \{
\text{n <- x@nbits}
\text{z <- qsimulatR::qft(x)}
\text{for(j in c(1:n))}{
\text{z <- Rtheta(bit=j, theta = 2*pi*y/2^(n-j+1)) * z}
}\text{z <- qft(z, inverse=TRUE)}
\text{return(invisible(z))}
\}
\]

Examples

\[
x <- \text{qstate(5, basis=as.character(seq(0, 2^5-1)))}
x
\]

\[
( 1 ) * 0
\]

\[
z <- \text{addbyqft(x, 3)}
z
\]

\[
( 1 ) * 3
\]

\[
z <- \text{addbyqft(z, 5)}
z
\]

\[
( 1 ) * 8
\]

\[
z <- \text{addbyqft(z, 30)}
z
\]

\[
( 1 ) * 6
\]

References