Christian B. Mendl, Martina Nibbi, Pedro Hack, Irene López Gutiérrez due: 15 May 2023, 08:00 on Moodle

Tutorial 3 (Number theory fundamentals[1])
We denote the *greatest common divisor* of two integers a and b by $\gcd(a, b)$. By the *representation theorem for the greatest common divisor*, $\gcd(a, b)$ is the least positive integer which can be written in the form $ax + by$, where x and y are integers. Two integers a and b are said to be *co-prime* if their greatest common divisor is 1.

Euclid's algorithm can efficiently compute $\gcd(a, b)$. It is based on the following

Theorem. *Let a and b be integers, and let r be the remainder when a is divided by b. Then provided $r \neq 0$,*

$$\gcd(a, b) = \gcd(b, r).$$

As demonstration, we find $\gcd(6825, 1430)$ using Euclid's algorithm:

$$\begin{aligned} 6825 &= 4 \cdot 1430 + 1105 \\ 1430 &= 1 \cdot 1105 + 325 \\ 1105 &= 3 \cdot 325 + 130 \\ 325 &= 2 \cdot 130 + 65 \\ 130 &= 2 \cdot 65 \end{aligned}$$

From this we see that $\gcd(6825, 1430) = 65$.

Euclid's algorithm has runtime cost $\mathcal{O}(L^3)$, where L is the number of bits required to represent a and b. One can adapt it to compute the integers x and y in the representation

$$ax + by = \gcd(a, b),$$

too, by successive substitution. For the example above:

$$\begin{aligned} 65 &= 325 - 2 \cdot 130 \\ &= 325 - 2 \cdot (1105 - 3 \cdot 325) = -2 \cdot 1105 + 7 \cdot 325 \\ &= -2 \cdot 1105 + 7 \cdot (1430 - 1 \cdot 1105) = 7 \cdot 1430 - 9 \cdot 1105 \\ &= 7 \cdot 1430 - 9 \cdot (6825 - 4 \cdot 1430) = -9 \cdot 6825 + 37 \cdot 1430. \end{aligned}$$

(a) When does a number a have a multiplicative inverse in modular arithmetic, that is, given a and n, when does there exist an integer b such that $ab = 1 \mod n$? For example, $2 \cdot 3 = 1 \mod 5$, so the number 2 has multiplicative inverse 3 in arithmetic modulo 5. Answer this question by deriving the following

Proposition. *Let n be an integer greater than 1. Then another integer a has a multiplicative inverse modulo n if and only if $\gcd(a, n) = 1$, that is, a and n are co-prime.*

(b) Prove the following

Theorem (Fermat's little theorem)**.** *Suppose p is a prime, and a is any integer. Then $a^p = a \mod p$. If a is not divisible by p then $a^{p-1} = 1 \mod p$.*

This theorem has a generalization based on the Euler φ function: $\varphi(n)$ is defined to be the number of positive integers less than n which are co-prime to n. As an example, note that all positive integers less than a prime p are co-prime to p, and thus $\varphi(p) = p - 1$. One can derive the useful relation $\varphi(ab) = \varphi(a)\varphi(b)$ if a and b are co-prime via the Chinese remainder theorem.

(c) Deduce that, for p prime,

$$\varphi(p^\alpha) = p^{\alpha-1}(p - 1).$$

Thus, we can obtain $\varphi(n)$ based on the prime factorization of n, $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$:

$$\varphi(n) = \prod_{j=1}^{k} p_j^{\alpha_j - 1}(p_j - 1).$$

Finally, we state the following remarkable generalization of Fermat's little theorem, due to Euler:

Theorem. *Suppose a is co-prime to n. Then $a^{\varphi(n)} = 1 \mod n$.*

[1] M. A. Nielsen, I. L. Chuang: *Quantum Computation and Quantum Information.* Cambridge University Press (2010), Appendix 4

Exercise 3.1 (Order-finding)
Let x and N be positive integers with no common factors and $x < N$. Recall that the *order* of x modulo N is the least positive integer r such that $x^r = 1 \mod N$. We denote the number of bits required to represent N by L. The quantum algorithm for order-finding is the phase estimation algorithm applied to the unitary operator

$$U|y\rangle = \begin{cases} |x \cdot y \mod N\rangle & 0 \le y < N \\ |y\rangle & N \le y < 2^L \end{cases}$$

for $y \in \{0, 1, \dots, 2^L - 1\}$. (Only the case $y < N$ is relevant here.)

(a) We discuss in the lecture that the states

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \mathrm{e}^{-2\pi i s k / r} |x^k \mod N\rangle \quad \text{for } s = 0, 1, \dots, r-1$$

are eigenstates of U with corresponding eigenvalues $\mathrm{e}^{2\pi i s/r}$, and that

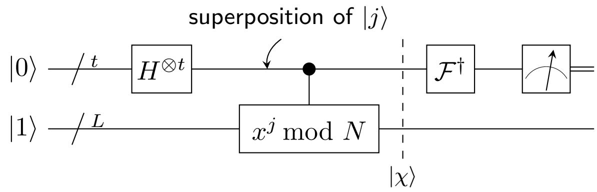
$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle . \tag{1}$$

Verify the following generalization of Eq. (1):

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \mathrm{e}^{2\pi i s k / r} |u_s\rangle = |x^k \mod N\rangle \quad \text{for all } k = 0, 1, \dots, r-1.$$

Hint: Use that $\frac{1}{r} \sum_{s=0}^{r-1} \mathrm{e}^{2\pi i s k/r} = \delta_{0, k \bmod r}$ for all integer k.

Based on Eq. (1), the quantum algorithm for order-finding uses $|1\rangle$ as input in the second register. You should convince yourself that $U^j |1\rangle = |x^j \bmod N\rangle$. This leads to the following schematic circuit:



Thus the quantum state $|\chi\rangle$ before the inverse Fourier transform is

$$|\chi\rangle = \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t - 1} |j\rangle |x^j \mod N\rangle .$$

(b) In the following, we set $N = 15$ and $x = 7$. What is the order r of x modulo N? Write down the state $|\chi\rangle$ explicitly for $t = 5$.

The *principle of implicit measurement* states that, without loss of generality, any unterminated quantum wires (qubits which are not measured) at the end of a quantum circuit may be assumed to be measured.

(c) Apply this principle by projecting $|\chi\rangle$ from (b) onto one of the (randomly selected) basis states appearing in the second register, say $|4\rangle$: that is, retain only basis states of the form $|j\rangle |4\rangle$ in $|\chi\rangle$, and normalize the resulting state $|\chi'\rangle$ to 1.

(d) Finally, compute the inverse Fourier transform $\mathcal{F}^\dagger |\chi'\rangle$, and plot the probability distribution of the result.
Hint: You can use the following Python code for this purpose, where you still have to insert $|\chi'\rangle$ represented as vector. Because of different conventions, we use NumPy's forward Fourier transform here.

```
import numpy as np
import matplotlib.pyplot as plt

chip = np.array([...])

Fchip = np.fft.fft(chip, norm='ortho')

plt.plot(np.abs(Fchip)**2, '.')
```

The nonzero entries of $\mathcal{F}^\dagger |\chi'\rangle$ should appear at indices ℓ with $\frac{\ell}{2^t} = \frac{s}{r}$ for some $s \in \{0, 1, \dots, r-1\}$, in accordance with phase estimation.

3.1. (a)

$$\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1} e^{2\pi i s k/r}|u_s\rangle = \frac{1}{\sqrt{r}}\cdot\frac{1}{\sqrt{r}}\sum_{s=0}^{r-1}\sum_{k=0}^{r-1} e^{2\pi i s j/r - 2\pi i s k/r}|x^k \bmod N\rangle = \frac{1}{r}\cdot\sum_{s,k}^{r-1}\exp[2\pi i(sj-sk)/r]\cdot|x^k \bmod N\rangle$$

$$= \frac{1}{r}\cdot\sum_{s,k}\exp(2\pi i s(j-k)/r)\cdot|x^k \bmod N\rangle \Longrightarrow \sum_{k=0}^{r-1}\delta_{0,(j-k)\bmod r}\cdot|x^k \bmod N\rangle =$$

based on hint: $\frac{1}{r}\sum_{s=0}^{r-1}\exp(2\pi i s k/r) = \delta_{0,k \bmod r}$

if $(j-k)\bmod r = 0 \Rightarrow \sum_{k=0}^{r-1}\delta_{0,(j-k)\bmod r}|x^k \bmod N\rangle = |x^k \bmod N\rangle$

(b)

$N = 15$

$x = 7$

r = order of $x \bmod N$; least pos int. $x^r = 1 \bmod N$

$7^r = 1 \bmod 15$

order r of $x \bmod N$ ($7 \bmod 15$)

$7^r = 1 \bmod 15$

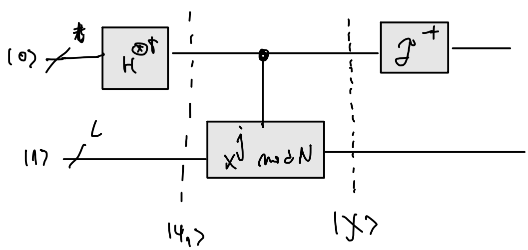
$t = 5$

$N = 2^t$

$$\left.\begin{array}{ll} r=0 & 7^0 = 1 \bmod 15\\ 1 & 7^1 = 7 \bmod 15\\ 2 & 7^2 = 4 \bmod 15\\ 3 & 7^3 = 13 \bmod 15\\ 4 & 7^4 = 1 \bmod 15\end{array}\right\} x^j \bmod N$$

$\Rightarrow$ the order r of $x \bmod N$ is 4

$|\psi_1\rangle = H|0\rangle \otimes |1\rangle^{\otimes L}$



$L = 4$

$$|\psi_1\rangle = H^{\otimes 5}|0\rangle \otimes |1\rangle^{\otimes L}$$

$$= \frac{1}{\sqrt{2^t}}\sum_{j=0}^{2^t-1}|j\rangle|x^j \bmod N\rangle \qquad t = 5$$

$$= \frac{1}{\sqrt{2^t}}\sum_{j=0}^{2^t-1}|j\rangle|x^j \bmod N\rangle = \frac{1}{\sqrt{32}}\sum_{j=0}^{31}|j\rangle|x^j \bmod N\rangle = \frac{1}{\sqrt{32}}\Big[(|0\rangle+|4\rangle+|8\rangle+\dots+|28\rangle)|1\rangle + (|1\rangle+|5\rangle+\dots+|29\rangle)|7\rangle + (|2\rangle+|6\rangle+\dots+|30\rangle)|4\rangle + (|3\rangle+|7\rangle+\dots+|31\rangle)|13\rangle\Big]$$

(c). Retaining basis states $|j\rangle|k\rangle$; $|y\rangle$

$$|x'\rangle = \frac{1}{\sqrt{8}}\left[\,|2\rangle + |6\rangle + |10\rangle + |14\rangle + |18\rangle + |22\rangle + |26\rangle + |30\rangle\,\right]$$

(d) Compute the inverse Fourier transform $\mathcal{F}^{\dagger}|x'\rangle$ and plot...

```python
import numpy as np
import matplotlib.pyplot as plt
```

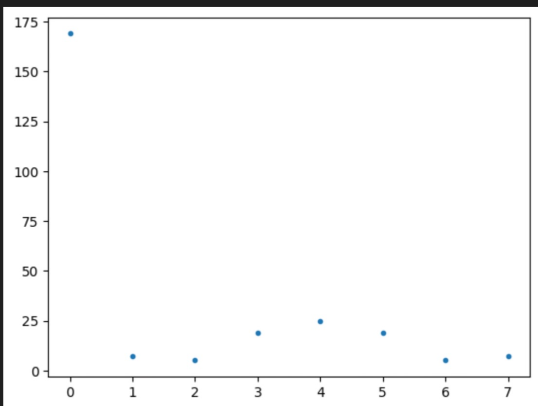
✓ 0.0s Python

```python
chip = 1/(np.sqrt(8))*np.array([2, 6, 10,14,18,22,2,30])

Fchip = np.fft.fft(chip, norm='ortho')
plt.plot(np.abs(Fchip)**2, '.')
```

✓ 0.2s Python

```
[<matplotlib.lines.Line2D at 0x265ffd0b070>]
```



Exercise 3.2 (Quantum Fourier transform for prime dimensions)
Construct a quantum circuit which performs the quantum Fourier transform

$$|j\rangle \mapsto \frac{1}{\sqrt{p}} \sum_{k=0}^{p-1} \mathrm{e}^{2\pi i jk/p} |k\rangle ,$$

where p is a prime number.

Hints: You can assume that $|j\rangle$ is encoded via n qubits, with n the smallest integer such that $p \leq 2^n$, and that $j < p$. To simplify the problem, you can further assume that the output of the circuit is to be stored in a separate quantum register with n qubits, which is already initialized to the equal superposition state $|\Phi_p\rangle = \frac{1}{\sqrt{p}} \sum_{k=0}^{p-1} |k\rangle$. Thus the overall transformation reads

$$|j\rangle \otimes |\Phi_p\rangle \mapsto |j\rangle \otimes \mathsf{QFT}\,|j\rangle = |j\rangle \otimes \left(\frac{1}{\sqrt{p}} \sum_{k=0}^{p-1} \mathrm{e}^{2\pi i jk/p} |k\rangle \right) .$$

(Note that the two registers become entangled in general.) Finally, you may use (controlled) rotation gates of the form

$$R_{m,p} = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{2\pi i\, 2^m/p} \end{pmatrix} .$$

Quantum Fourier transform for prime dimensions.

sum: $j\,{}^{k}/_{p}$

encoding bit $\rightarrow \frac{jk}{p} = \frac{2^m}{p}$

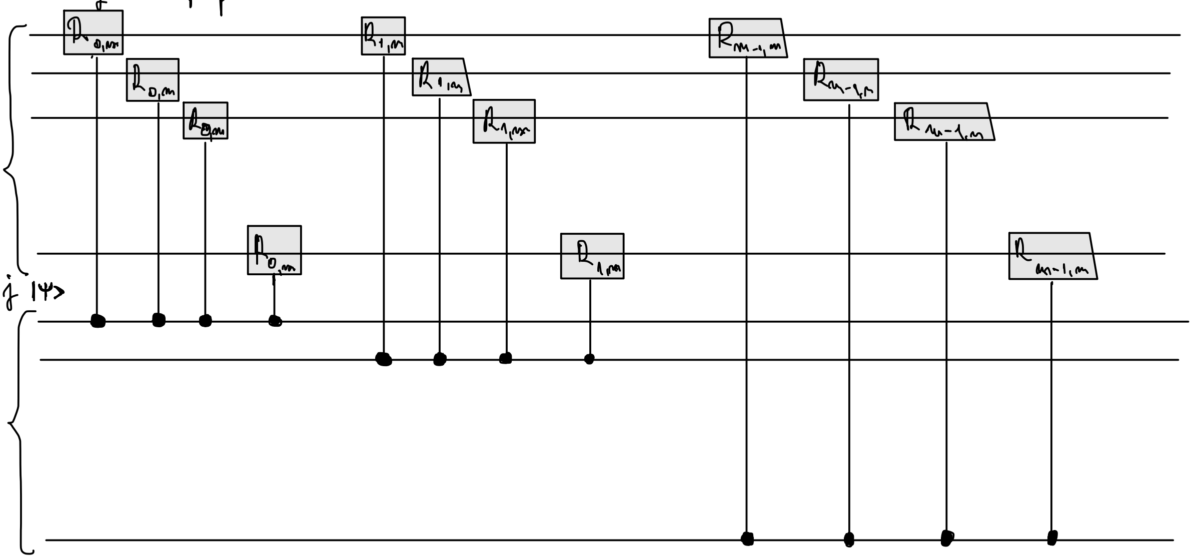
$$R_{m,p} = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i 2^m/p} \end{pmatrix}$$

$$\frac{1+2+\cdots p-1}{p}$$

final state: $|j\rangle = \frac{1}{\sqrt{p}} \sum_{k=0}^{p-1} e^{2\pi i j k/p} |k\rangle$ $\qquad |0\rangle = \frac{1}{\sqrt{p}} \cdot \sum_{k=0}^{p-1} |k\rangle$

$$\frac{1}{p} + \frac{2}{p} + \cdots \frac{p-1}{p}$$

$|\Phi_p\rangle$ equal superposition $\qquad$ note: $m = p$



$$R_{m,p} = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i 2^m/p} \end{pmatrix}$$