6. Quantum computing

6.1 some elements of classical computing

Key elements of modern computers go back to
Charles Babbage (1791–1871)
"founding father" of formal theory of computing
Alan Turing (1912–1954)

calculation: systematic generation of a finite set of (binary) symbols with abstract meaning from a different finite set of (binary) symbols

is equivalent to

(1) $f: \{0,1\}^n \longrightarrow \{0,1\}^m$

can be reduced to a <u>decision problem</u> 0 = NO, 1 = YES

(2) $f: \{0,1\}^n \longrightarrow \{0,1\}$

total number of possible functions is 2^{2^n}
2 outputs and 2^n inputs. Each of these 2^{2^n} functions can be decomposed into a <u>sequence of elementary logic operations</u>

$\Rightarrow$ (A) <u>network-model of computation</u>

input

(3) $\quad x = x_1 x_2 \cdots x_n \qquad x_i \in \{0,1\}$

$X^{(a)}$: set of all x for which $f = 1 \qquad a = 1, 2, 3 \ldots m$

(4) $$f^{(a)}(x) = \begin{cases} 1 & \text{if } x \in X^{(a)} \\ 0 & \text{otherwise} \end{cases}$$

then

(5) $$f(x) = f^{(1)}(x) \vee f^{(2)}(x) \vee f^{(3)}(x) \vee \cdots f^{(m)}(x)$$

($\vee$ = logic OR); so we have to calculate functions $f^{(a)}(x)$

If $x = 111\ldots1 \in X^{(a)}$ then we can write

(6) $$f^{(a)}(x) = x_1 \wedge x_2 \wedge x_3 \wedge \cdots$$

($\wedge$ = logic AND). If $x \in X^{(a)}$ contains a "0" at a certain place i we can represent this by a negation $\neg x_i$ ($\neg$ = logic NOT)

(7) e.g. $\quad f^{(a)}(x) = (\neg x_1) \wedge x_2 \wedge x_3 \wedge (\neg x_4) \wedge \cdots$

etc.

In order to evaluate (5) we need an m-fold copy of the input x

(8) $\quad$ COPY: $\quad x \longrightarrow xx$

This shows that in order to evaluate an

arbitrary function $f: \{0,1\}^n \to \{0,1\}$ we need a finite number of four elementary logic operations

(9) COPY, NOT (1-bit (input)) AND, OR (2-bit (input))

If one can prepare a cbit in a certain state ($x=0$ or $x=1$) it is sufficient to have one 2-bit operation U and NOT as 1-bit operation

(10) $(x,y) \xrightarrow{U} (1-x, 1-xy)$

is equivalent to NAND (NOT AND) if we ignore the first output bit. For $y=1$ U is equivalent to copy if we perform additionally a bit-wise NOT.

(11) $(x,1) \xrightarrow{U} (1-x, 1-x) \xrightarrow{\text{bitwise NOT}} (x,x) \hat{=} \text{COPY}$

U is called a universal 2-bit gate

network model of classical computing

- contains few basis elements (1-bit, 2-bit gates)
- requires preparation of bits in certain states

question: How many elementary operations are needed to evaluate f when size (n) of input $X = x_1 x_2 \cdots x_n$ is increased?

(B) complexity classes of classical computation

We consider evaluation of $f: \{0,1\}^n \to \{0,1\}$ as simple if the number of elementary gates is a polynomial in n. All of such functions (problems) are said to belong to the complexity class

$P = \{$ decision problems which can be solved with poly(n) elements $\}$

All functions that do not belong to this class correspond to difficult or hard problems

The class of problems for which the verification of the solution is simple in the above sense is the class

$NP = \{$ decision problems for which the verification of a solution requires poly(n) elements $\}$

Note: The problem itself may be simple or hard obviously

(12) $\quad P \subseteq NP$

(13) conjecture $\quad P \neq NP$

example from NP : CIRCUT-SAT

(14) $$C[f(x)] = \begin{cases} 1 & \text{if } x \text{ exists such that } f(x)=1 \\ 0 & \text{else} \end{cases}$$

theorem of Cook (1971):

Every problem from class NP can be reduced to CIRCUIT-SAT with polynomial effort

if CIRCUIT-SAT $\in$ P $\Rightarrow$ NP = P

there is a whole class of problems from NP that have the same property as CIRCUIT-SAT, i.e. that all problems from NP can be reduced with polynomial effort to them!

NPC : NP-complete problems

(15) if $NP \neq P \Rightarrow NPC \cap P = \emptyset$

$NPC \cup P \neq NP$

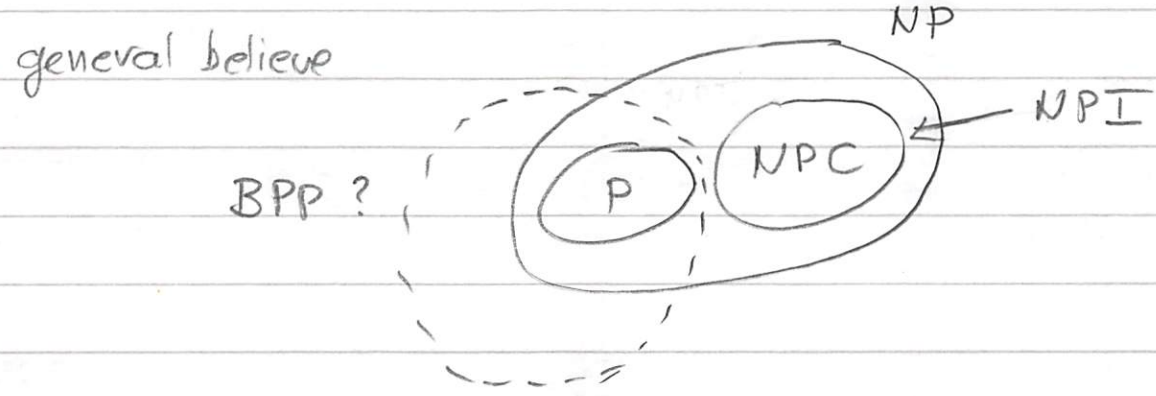
i.e. if NP ≠ P then there are problems in NP which are neither NPC nor P, called NPI. One of these problems is believed to be the factorization into prime numbers

So far we have considered deterministic computations. It turns out however that it can be more efficient to use an algorithmic calculation path that has built-in elements of randomness. Such a probabilistic

evaluation of a decision problem gives a correct result only with finite probability. If the correct decision is found with probability $p = \frac{1}{2} + \delta$, $\delta > 0$ the problem can be solved by repetition and majority vote with arbitrary precision. The class of problems that can be solved with probabilistic calculations in polynomial effort and bounded error probability is called

$$BPP = \{ \text{bounded-error probabilistic polynomial} \}$$

The general relation between these classes is not yet fully known.



rem: there are decision problems which cannot be solved (with finite effort) (Turing, Goedel)
example: stopping problem

universal computer: Turing machine (not discussed here)

(c) reversible computing

To generalize to quantum computing where elementary operations are unitary operations we have to consider the concept of reversible computation. A second argument to consider reversible processes is that irreversible ones lead to entropy production and thus heat.

reversible computation $\hat{=}$ invertible functions

(16) $$f: \{0,1\}^n \longrightarrow \{0,1\}^n$$

every function (1) can be turned into a symmetric one by addition of ancillas. consider

(17) $$f(x): \{0,1\}^n \to \{0,1\}^m$$

(18) $$\tilde{f}(x): \{0,1\}^{n+m} \to \{0,1\}^{n+m}$$

where

(19) $$\tilde{f}(x, \underbrace{0,0,\ldots 0}_{m}) = (x, f(x))$$

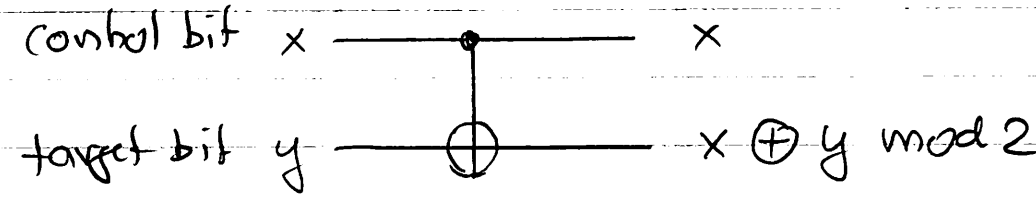
Of all possible 1-bit gates two are reversible

IDENTITY NOT

Of all 256 (2^8) 2-bit gates are 24 reversible

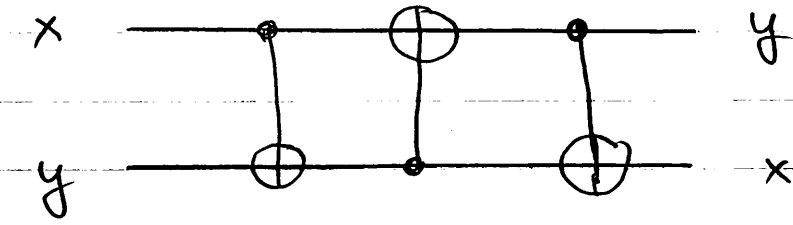
e.g.

CNOT: $(x,y) \longrightarrow (x, x \oplus y \bmod 2)$



CNOT ≅ NOT for $x = 1$ CNOT ≅ COPY for $y = 0$

bit-swapping



reversible 1-bit and 2-bit gates are not universal since they are all linear

$$(20) \quad \begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow \begin{pmatrix} x' \\ y' \end{pmatrix} = \underline{\underline{M}} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix}$$

$$\begin{pmatrix} a \\ b \end{pmatrix} \in \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$M \in \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \right\}$$

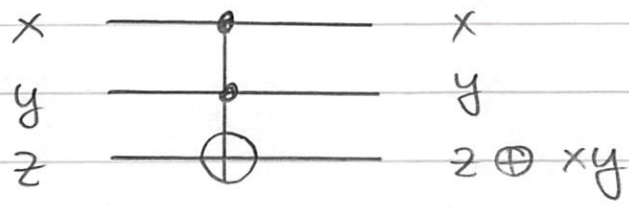
M^{-1} exists

total $6 \times 4 = 24$

repetition of linear operations leads again to a linear operation ⇒ non-linear functions cannot be calculated.

For $n \geq 3$ reversible gates exist that correspond to a nonlinear mapping.

Toffoli-gate controlled-controlled-NOT



Toffoli gate is universal if preparation of input bits in certain states (0 or 1) is possible

6.2 quantum network computer

input → string of classical bits

1 0 0 1 1 0 1 0 1

↓

factorized state of qubits

$|1\rangle|0\rangle|0\rangle|1\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle$

gates → unitary operations

output → von-Neumann projection of set of qubits to computational basis $\{|0\rangle, |1\rangle\}$

↓

string of classical bits

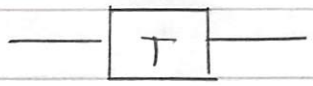
0 1 1 0 1 0 1 1 0

important 1-qubit gate

Hadamard

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$\pi/8$

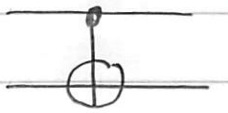


$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$$

combination of Hadamard and $\pi/8$ gate allows to approximate any unitary operation on a single qbit with arbitrary precision

2-qubit gate

CNOT

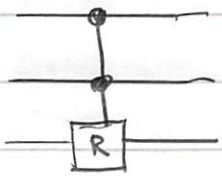


$$\begin{array}{c} |00\rangle\ |01\rangle\ |10\rangle\ |11\rangle \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

any 2-qubit gate can be decomposed into a C-NOT and a 1-qubit gate

3-qubit gate

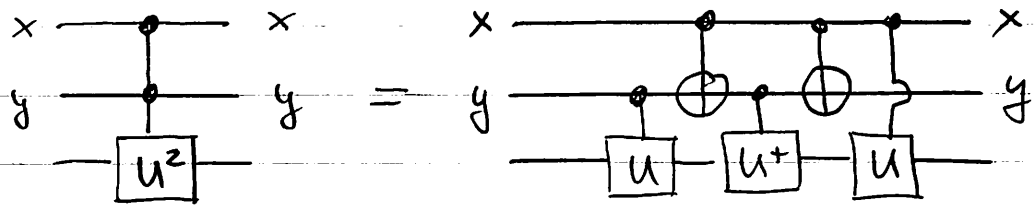
David Deutsch has shown (1989) that in analogy to the classical Toffoli gate a universal 3-qubit gate exists



$$R = -i \exp\left\{ i \frac{\theta}{2} \sigma_x \right\}$$

if 1st and 2nd qubit in state $|1\rangle$

can be decomposed into 2-qubit gates



action on 3rd qubit

$$U^y U^{+(x \oplus y)} U^x = U^{y - x \oplus y + x} = U^{y-(x+y-2xy)+x}$$

$$= U^{2xy} \quad \Rightarrow \quad U^2 = R(\theta)$$

theorem of Deutsch (1985)

> Any d-dimensional unitary matrix can be decomposed into $2d^2 - d$ unitary 2×2 matrices

remark: since the 2×2 matrices can also act in the space of two different qubits one- and two-qubit gates are needed

To realize an arbitrary unitary U of dimension d requires poly(d) 1-qubit gates that act only on the 2-dimensional subspaces $\mathcal{H}_i$ of individual qubits and poly(d) 2-qubit gates that act on the 4-dimensional subspaces $\mathcal{H}_i \otimes \mathcal{H}_j$ of two arbitrary qubits

$\Rightarrow$ network model of quantum computation

6.3 quantum algorithms

calculations on a quantum computer will in general be of probabilistic nature. Therefore we now define a class

$BQP = \{$ problems that can be solved with bounded error on a quantum computer with polynomial effort $\}$

We know that

(21) $$P \subseteq BPP \subseteq BQP$$

but we conjecture

(22) $$BQP \neq BPP$$

we will now discuss 2 classes of quantum algorithms

(i) non-exponential speed-up (Grover search algorithm)
(ii) exponential speed-up of potential hard problems

(A) Deutsch and Deutsch-Jozsa problem

does not belong to the above mentioned classes but is instructive to see where speedup could emerge in a quantum system

geg: $f: \{0,1\} \longrightarrow \{0,1\}$

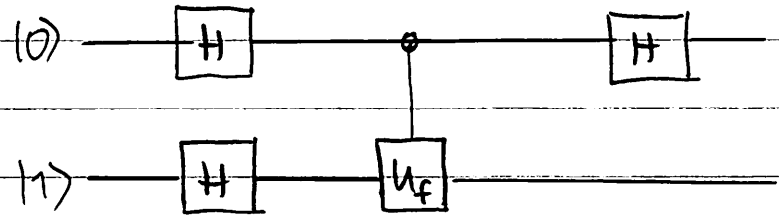
$f(0)=0$	or	$f(0)=1$	f: constant
$f(1)=0$		$f(1)=1$	

(23)

$f(0)=0$	or	$f(0)=1$	f: balanced
$f(1)=1$		$f(1)=0$	

In order to decide if f belongs to the "constant" or "balanced" class two inquiries are needed in a classical (deterministic) calculation. Quantum mechanically this can be achieved within one inquiry

(24) $f \rightarrow \hat{U}_f : |x\rangle|y\rangle \longrightarrow |x\rangle|y \oplus f(x)\rangle$



algorithm

(25) $|0\rangle|1\rangle \xrightarrow{\text{H-gates}} \frac{1}{2}(|0\rangle+|1\rangle)(|0\rangle-|1\rangle) = \frac{1}{2}\sum_{x=0}^{1}|x\rangle(|0\rangle-|1\rangle)$

(26) $U_f|x\rangle(|0\rangle-|1\rangle) = \begin{cases} |x\rangle(|0\rangle-|1\rangle) & \text{if } f(x)=0 \\ |x\rangle(|1\rangle-|0\rangle) & f(x)=1 \end{cases}$

$= (-1)^{f(x)}|x\rangle(|0\rangle-|1\rangle)$

$$\frac{1}{2}\sum_{x=0}^{1}|x\rangle(|0\rangle-|1\rangle) \xrightarrow{U_f} \frac{1}{2}\sum_{x=0}^{1}(-1)^{f(x)}|x\rangle(|0\rangle-|1\rangle)$$

(27)
$$= \pm\frac{1}{2}\begin{cases}(|0\rangle+|1\rangle)(|0\rangle-|1\rangle) & f: \text{constant}\\ (|0\rangle-|1\rangle)(|0\rangle-|1\rangle) & f: \text{balanced}\end{cases}$$

measurement of 1st qubit $\Rightarrow$ f balanced or constant

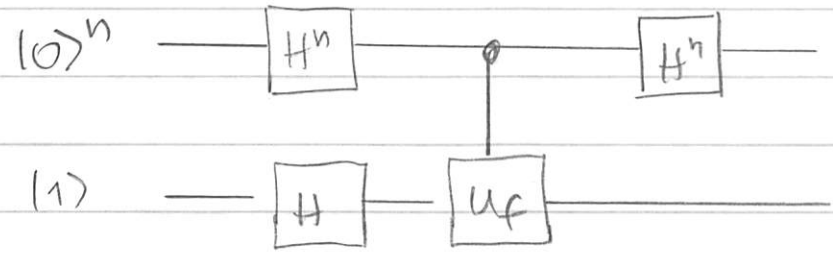
essence of this speedup as compared to classical case:
use of superposition states

Speedup by superposition: quantum parallelism

effect is even more dramatic if we consider a function

(28) $f: \{0,1\}^n \longrightarrow \{0,1\}$ (Deutsch-Jozsa)

let f be either constant, i.e. has the same output 0 or 1 for all inputs or balanced i.e. it has the same number of 0 and 1 in the output. To decide classically if f is constant or balanced $2^{n-1}+1$ inquiries are needed. Quantum mechanically a single one is sufficient



Here $H^n = \underbrace{H \otimes H \otimes \cdots \otimes H}_{n}$ is the bitwise Hadamard

$$H^n |x\rangle = \prod_{j=1}^{n} \frac{1}{\sqrt{2}} \sum_{y_j=0}^{1} (-1)^{x_j y_j} |y_j\rangle$$

(29)
$$= \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle$$

$x \cdot y$ is bitwise AND mod 2.

gate action

$$|0\rangle^n |1\rangle \xrightarrow{H^n, H} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$$

$$\xrightarrow{U_f} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$$

(30)
$$\xrightarrow{H^n} \frac{1}{2^n} \sum_{y=0}^{2^n-1} \underbrace{\sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot y}}_{= S(y)} |y\rangle \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$$

f constant

(31)
$$S(y) = \pm \sum_{x=0}^{2^{n-1}} (-1)^{x \cdot y} = \pm \delta_{y,0} \cdot 2^n$$

$\pm$: global sign

$\delta_{y,0}$: (if $y \neq 0$ there are equal numbers of terms "+1" and "−1")

$\Rightarrow$ output state $|y=0\rangle = |0^n\rangle$

f balanced

$$(32) \qquad \delta(y=0) = \pm \sum_{x=0}^{2^n-1} (-1)^{f(x)} = 0$$

i.e. if f balanced $\Rightarrow$ output state $|y=0\rangle$ does not occur

$\Rightarrow$ projection of $|y\rangle$ to $|0\rangle$ sufficient to find wether f constant or balanced

The Deutsch-Josza problem is however a problem inside BPP. Assume we want to determine classification of f with probability

$$p \geq 1-\varepsilon$$

If f is balanced the probability to obtain the same result in k random evaluations is $p_t = 2^{-(k-1)}$. Thus if k successive evaluations give indeed the same result (if they don't then we know anyway) the probability that f is nevertheless balanced is

$$(33) \qquad p_b = \frac{1}{2^{k-1}+1}$$

We of course will rather assume it is constant and this is correct with probability

$$(34) \qquad p = 1-\varepsilon \qquad \varepsilon = \frac{1}{2^{k-1}+1}$$

Thus to reach error ε one needs $k \sim \frac{1}{2}\log\left(\frac{1}{\varepsilon}\right)$ trials.

(B) Simon problem

given

(35) $$f: \{0,1\}^n \longrightarrow \{0,1\}^n$$

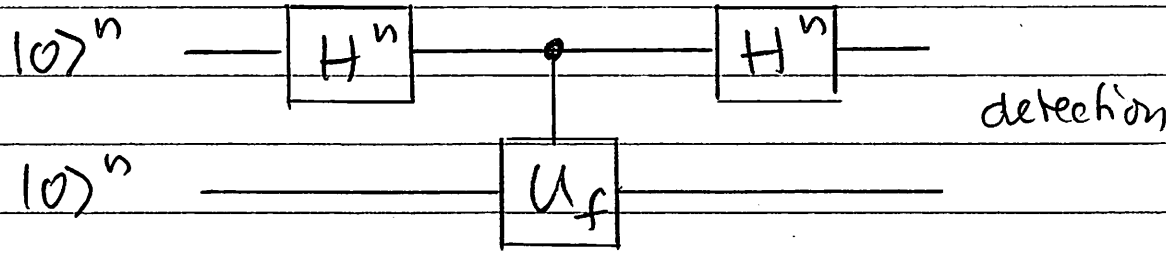
which has a (binary) period a, i.e.

(36) $$f(x) = f(y) \quad \text{if} \quad y = x \oplus a$$

$\oplus$ is bitwise addition mod 2. The task is to find a.

- Classically this is a hard problem (NP). We have to test pairs x, y such that $x \oplus y = a$

- quantum mechanically



short notation $|\underline{0}\rangle = |0\rangle^n = |0\rangle|0\rangle \cdots |0\rangle$

(37)
$$|\underline{0}\rangle|\underline{0}\rangle \xrightarrow{H^n} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle|0\rangle$$

$$\xrightarrow{U_f} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle|f(x)\rangle$$

- measure 2nd register: yields randomly one of the 2^n possible outputs of f. Suppose measurement yielded $f(x_0)$. Since both x_0 and $x_0 \oplus a$ give the same result, the first register is in the state

(38) $$\frac{1}{\sqrt{2}} \left(|x_0\rangle + |x_0 \oplus a\rangle \right)$$

after the measurement.

- to determine a: bitwise Hadamard of first register

$$H^n \frac{1}{\sqrt{2}} \left(|x_0\rangle + |x_0 \oplus a\rangle \right) =$$

$$= \frac{1}{2^{(n+1)/2}} \sum_{y=0}^{2^n-1} \left[(-1)^{x_0 \cdot y} + (-1)^{(x_0 \oplus a) \cdot y} \right] |y\rangle$$

(39) $$= \frac{1}{2^{(n+1)/2}} \sum_{y=0}^{2^n-1} (-1)^{x_0 \cdot y} \left[1 + (-1)^{a \cdot y} \right] |y\rangle$$

$$= \frac{1}{2^{(n+1)/2}} \sum_{a \cdot y = 0} (-1)^{x_0 \cdot y} |y\rangle$$

measurement of $|y\rangle$-register projects to one of the 2^{n-1} states with $a \cdot y = 0$

- repeat protocol until we have n linear independent values y_i with

(40) $y_i \cdot a = 0 \qquad i = 1, 2, \ldots, n$

this is a set of n linear equations from which we can obtain the n-bit word a

In order to find n linearly independent values y_i the algorithm has to be performed on average $O(n)$ times.

Simon problem (which is a so-called oracle) shows

$$BQP \neq BPP$$

(C) Grover search algorithm

problem: given an unsorted list with N entries how to find a certain entry? (reverse telephone book). classically N/2 steps are needed

rem: search in ordered list requires $O(\log N)$ steps (successive bi-section)

Lev Grover: algorithm $N \to \sqrt{N}$

- does not change complexity class (NP) but is a speedup

coding of "telephone-book" with the help of an oracle. For a given "telephone number" ω the

corresponding value is x_0. The oracle is now defined as

(41) $$f_w(x) = 1 \quad \text{for } x = x_0$$
$$f_w(x) = 0 \quad \text{else}$$

quantum algorithm $f \to U_f$ ($x \to f(x)$ easy)

(42) $$U_w : |x\rangle|y\rangle \longrightarrow |x\rangle|y \oplus f_w(x)\rangle$$

As seen before

(43) $$U_w |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \longrightarrow (-1)^{f_w(x)} |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Ignoring the second register, the action of U_w is

(44) $$U_w |x\rangle \longrightarrow (-1)^{f_w(x)} |x\rangle$$

I.e. U_w leads to a reflection (sign change) of states with respect to a hypersurface orthogonal to $|x_0\rangle$

(45) $$U_w \hat{=} \mathbb{1} - 2|x_0\rangle\langle x_0|$$

Now we can use the superposition principle of quantum mechanics. We choose an initial state

(46) $$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

$|s\rangle$ contains the state $|x_0\rangle$ which we are looking for with probability $1/N$. Grovers Algorithm aims at an iterative amplification of this probability by application of

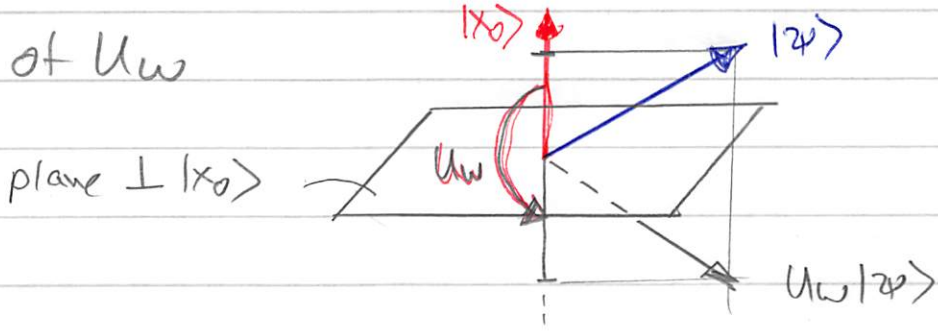
(47) $$R = U_s U_w \qquad U_s = 2|s\rangle\langle s| - 1 = -(1 - 2|s\rangle\langle s|)$$

"reflection about the mean"

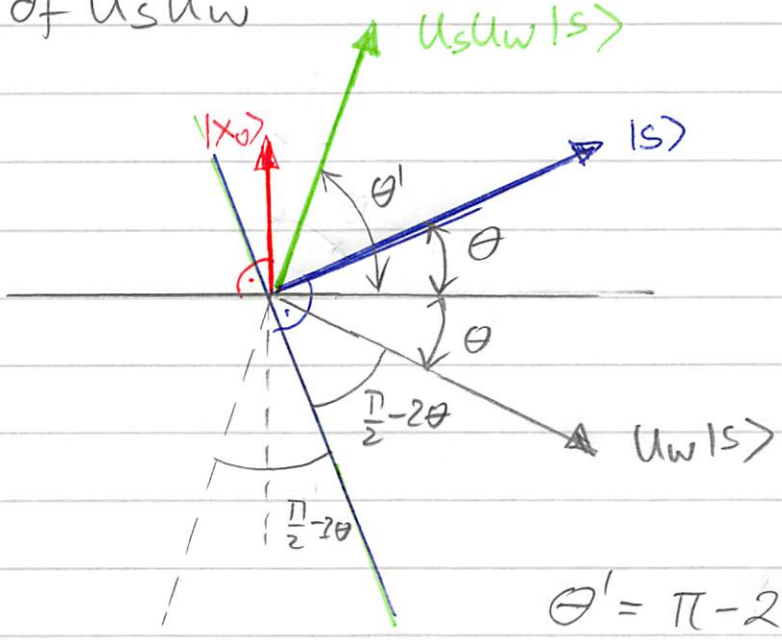
geometric illustration

(48) $$|\langle s|x_0\rangle| = \frac{1}{\sqrt{N}} = \sin\theta \approx \theta$$

action of U_w



action of $U_s U_w$



(49) $$\theta' = \pi - 2\left(\frac{\pi}{2} - 2\theta\right) - \theta = 3\theta$$

⇒ in every iteration step the vector is shifted by $\Delta\theta = 2\theta$ in the direction of $|x_0\rangle$

(50) $$|\psi_0\rangle = |s\rangle \qquad |\psi_{k+1}\rangle = R\,|\psi_k\rangle$$

thus

(51) $$|\psi_k\rangle = \alpha_n \sum_{x \neq x_0} |x\rangle + \beta_n |x_0\rangle$$

(52) $$\alpha_k = \frac{1}{\sqrt{N-1}} \cos((2k+1)\theta) \qquad \beta_n = \sin((2k+1)\theta)$$

amplitude of state which we are looking for increases as long as $(2k+1)\theta \leq \pi/2$. Optimum number of steps is

(53) $$(2T+1)\theta \simeq \frac{\pi}{2}$$

for large N: $\sin\theta \simeq \theta = \frac{1}{\sqrt{N}}$ ⇒

(54) $$T = \frac{\pi}{4}\sqrt{N} + \mathcal{O}(1)$$

i.e. after $\sqrt{N}$ steps $|x_0\rangle$ is approached with probability close to 1

search in unsorted list successful in $\sqrt{N}$ steps

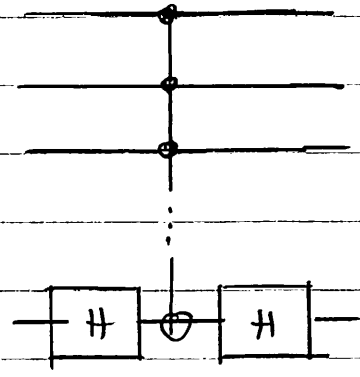
implementation of U_s and initialization

(55) $$|s\rangle = H^{(n)} |0\rangle$$

i.e.

(56) $$U_s = 2|s\rangle\langle s| - 1 = H^{(n)} (2|0\rangle\langle 0| - 1) H^{(n)}$$

implementation of $|1111\ldots1\rangle \longrightarrow -|111111\ldots\rangle$
while other states unaffected



this gate can be realized by $O(n)$ elementary gates
i.e. Grovers algorithm requires $\sqrt{N}$ poly $(n = \log N)$ gate operations

one can show that Grovers algorithm is optimal, i.e. there is no faster way to search an unsorted list

(D) Quantum fourier transform and Shors factorization algorithm

given a function f

(57) $$f: \{0,1\}^n \longrightarrow \{0,1\}^n$$

with an unknown period r, $1 \ll r \ll 2^n$, i.e.

(58) $$f(x) = f(x+mr) \qquad r, m \in \mathbb{N}$$
$$x, x+mr \in \{0,1,2,\dots,2^n-1\}$$

To find r is classically a difficult problem even if evaluation of $f(x)$ can be made in poly(n) steps.

The key element of a quantum algorithm to find the period is the quantum Fourier transform

(59) $$QFT: \quad |x\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i \frac{xy}{N}} |y\rangle$$

with $N = 2^n$. Assume for now that we can implement QFT in an efficient way. How can we find the period r of an unknown function f with this?

Analogously to Simon problem $\quad |\underline{0}\rangle = \underbrace{|0\rangle|0\rangle\cdots|0\rangle}_{n}$

(60) $$|\underline{0}\rangle\,|\underline{0}\rangle \xrightarrow{H^n} \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle\,|\underline{0}\rangle$$
$$\xrightarrow{U_f} \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle\,|f(x)\rangle$$

measurement of second register with result $f(x_0)$ and $0 \le x_0 < r$ projects first register to state

(61) $$\frac{1}{\sqrt{A}} \sum_{j=0}^{A-1} |x_0 + jr\rangle \qquad A \in \{0, 1, 2, \dots 2^n - 1\}$$

where

(62) $$N - r \le x_0 + (A-1)r < N$$

or

(63) $$A - 1 < \frac{N}{r} < A + 1$$

If N/r is integer $\Rightarrow A = N/r$

<u>problem:</u> How to find r from eq. (61)?
projection to computational basis does not give information about r $\Rightarrow$ QFT

(64) $$\text{QFT} \frac{1}{\sqrt{A}} \sum_{j=0}^{A-1} |x_0 + jr\rangle = \frac{1}{\sqrt{NA}} \sum_{y=0}^{N-1} e^{\frac{2\pi i x_0 y}{N}} \sum_{j=0}^{A-1} e^{\frac{2\pi i r y}{N}} |y\rangle$$

probability to find $|y\rangle$ in this state

(65) $$p(y) = \frac{A}{N} \left[\frac{1}{A} \sum_{j=0}^{A-1} e^{2\pi i \frac{jry}{N}} \right]^2$$ unknown x_0 does no longer appear

$p(y)$ is large if $\frac{ry}{N}$ integer number. Eg. if $\frac{N}{r} = A$ integer
$\Rightarrow$

(66) $$p(y) = \frac{1}{r} \left[\frac{1}{A} \sum_{j=0}^{A-1} e^{2\pi i \frac{jy}{A}} \right]^2 = \begin{cases} \frac{1}{r} & y = mA \\ 0 & \text{else} \end{cases}$$

in general

$$(67)\quad \sum_{j=0}^{A-1} e^{ij\theta_y} = \frac{e^{iA\theta_y}-1}{e^{i\theta_y}-1} \qquad \theta_y = \frac{2\pi r y \,(\mathrm{mod}\, N)}{N}$$

Let ask the question for which values of y (67) is large. Since $A-1 < \frac{N}{r}$ all terms of the sum in (67) are in the same half of the complex plain if

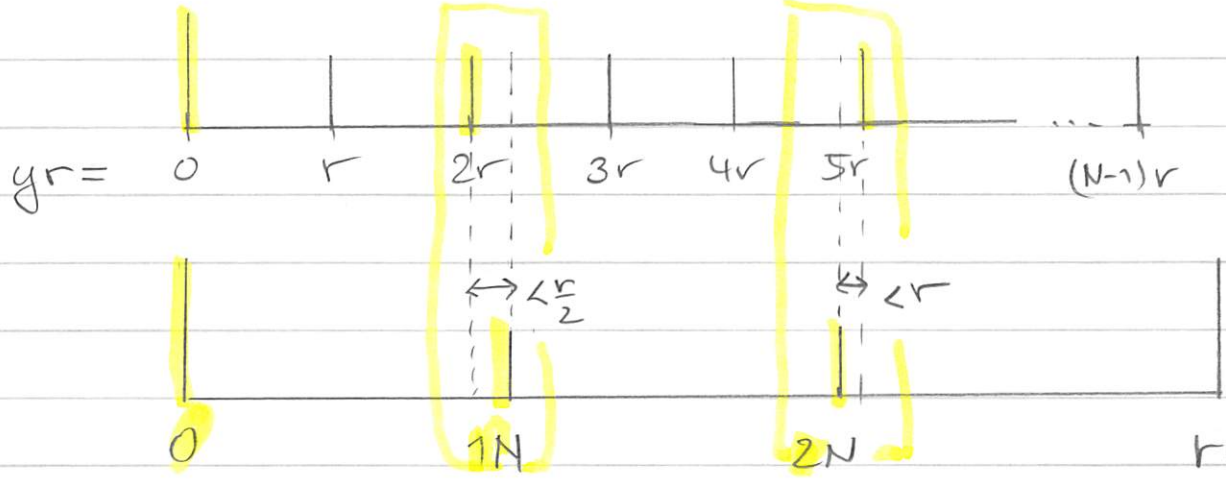
$$(68)\quad -\pi\frac{r}{N} \leq \theta_y \leq \pi\frac{r}{N}$$

In this case (67) is extremal. (68) is equivalent to

$$(69)\quad -\frac{r}{2} \leq yr \,(\mathrm{mod}\, N) \leq \frac{r}{2}$$

There are exactly r values of y in $\{0,1,\ldots,N-1\}$ which fulfill condition (69).

example

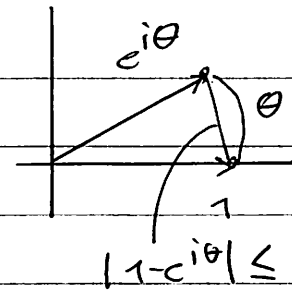


here $y=0$ $y=2$ $y=5$ ... fulfill (69)

$\Rightarrow$ 7 r values of y that fulfill (69)

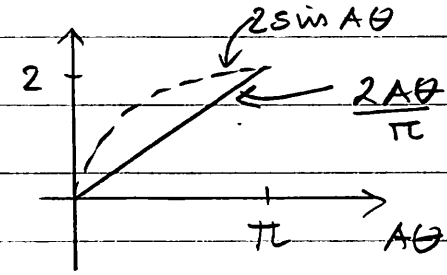
how

(70) $$|1-e^{i\theta}| \leq |\theta|$$



furthermore for $A|\theta| \leq \pi$

(71) $$|1-e^{iA\theta}| \geq \frac{2A|\theta|}{\pi}$$



Since $|1-e^{iA\theta}| = |e^{i\theta A/2}| 2\sin A\theta$

$= 2\sin A\theta$

this yields the estimate

(72) $$\left|\frac{e^{iA\theta}-1}{e^{i\theta}-1}\right| \geq \frac{2A|\theta|\,1}{\pi\,|\theta|} = \frac{2A}{\pi}$$

and thus

(73) $$P(y) \geq \frac{A}{N}\frac{4}{\pi^2} \geq \frac{4}{\pi^2}\left(\frac{1}{r}-\frac{1}{N}\right) \geq \frac{4}{\pi^2}\frac{1}{r}$$

for every value y that fulfills (63)

with probability $> \frac{4}{\pi^2}$

$\Rightarrow$ A measurement of the F-transformed state projects to one of the y-value that fulfill condition (69), i.e.

(74) $$k\frac{N}{r} - \frac{1}{2} \leq y \leq k\frac{N}{r} + \frac{1}{2}$$

where k is integer in $\{0,1,\ldots,r-1\}$

knowledge of y

(75) $$y \longrightarrow \frac{k}{r}$$

With high probability k and r have no common divisor, which allows us to determine r.

By evaluating U_f we can then easily check if

$$f(x) = f(x+r) \tag{76}$$

This is necessary since measurement can yield some other values of y that do not fulfill (69), however with small probability. If (76) is not fulfilled we can try neighboring values of y. If k and r do have a common divisor we throw everything away and start over again.

implementation of QFT

Let now discuss how QFT can be constructed using elementary gates.

$$\sum_{x=0}^{N-1} f(x)\,|x\rangle \longrightarrow \sum_{y=0}^{N-1} \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} e^{\frac{2\pi i x y}{N}} f(x)\,|y\rangle \tag{77}$$

corresponds to multiplication with $N \times N$ matrix $\left(e^{\frac{2\pi i}{N}}\right)^{xy}$ i.e. $2^n \times 2^n$ matrix. Naively one would expect that $O(N^2) = O(2^{2n})$ elementary operations needed (dim $\mathcal{H} = 2^n$). This is however not the case! On classical level there is the Fast Fourier Transform (FFT) that needs only $O(N \log N)$ steps.

binary representation of x, y

$$x = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \dots + x_1 2 + x_0$$
$$y = y_{n-1}2^{n-1} + y_{n-2}2^{n-2} + \dots + y_1 2 + y_0 \quad (78)$$

In order to evaluate $\left(e^{\frac{2\pi i}{N}}\right)^{x\cdot y}$ we can now ignore all terms in $x\cdot y$ that are multiples of $2^n = N$

$\Rightarrow$ for evaluating $(e^{2\pi i})^{x\cdot y/N}$

$$\frac{x\cdot y}{N} = \frac{x\cdot y}{2^n} \;\hat{=}\; y_{n-1}\frac{x_0}{2} + y_{n-2}\left(\frac{x_1}{2} + \frac{x_0}{2^2}\right)$$
$$+ \dots + y_0\left(\frac{x_{n-1}}{2} + \frac{x_{n-2}}{2^2} + \dots + \frac{x_0}{2^n}\right) \quad (79)$$

To evaluate this expression $O(n)$ steps are needed i.e. the FFT

$$\tilde{f}(x) = \frac{1}{\sqrt{N}} \sum_{y=1}^{N-1} e^{2\pi i \frac{xy}{N}} f(y) \quad (80)$$

can be evaluated for every of the N values x in $O(n = \log N)$ steps. Unfortunately the FFT only leads to a speedup from N^2 to $N \log N$ and thus does not changes the complexity class,

Using quantum parallelism we can however perform the FTrafo much more efficient! According to (78)

$$QFT|x\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i \frac{xy}{N}} |y\rangle = \quad (81)$$

$$= \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i [x_0]} |1\rangle\right)\left(|0\rangle + e^{2\pi i [x_0 x_1]} |1\rangle\right)$$
$$\cdots \left(|0\rangle + e^{2\pi i [x_0 x_1 \cdots x_{n-1}]} |1\rangle\right)$$

$N = 2^n$

where

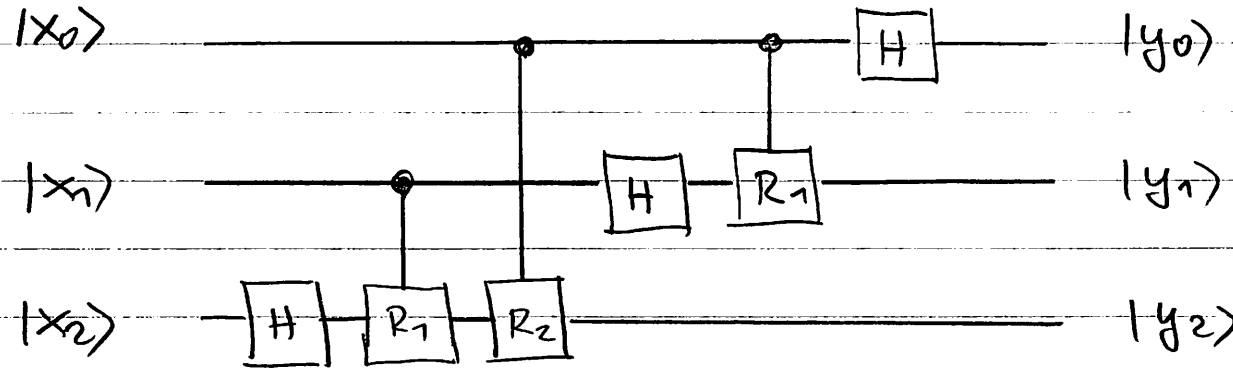
(82) $$[x_0 x_1 \cdots x_k] \equiv \frac{x_0}{2} + \frac{x_{k-1}}{2^2} + \cdots + \frac{x_0}{2^{k+1}}$$

Eq. (81) can efficiently be implemented

example: $n = 3$

$$x \longrightarrow x_0, x_1, x_2 \longrightarrow |x_0\rangle |x_1\rangle |x_2\rangle$$

binary repres.



etc.

where

(83) $$H|x_0\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i [x_0]} |1\rangle\right)$$

and

(84) $$R_d = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2^d} \end{pmatrix}$$

For QFT one Hadamard gate and $n(n-1)/2$ R-gates are required per input (qu)bit.
$\Rightarrow$

> A Quantumcomputer can determine the period of a function $f: Z_N \to Z_N$ in
>
> $$O((\log N)^2)$$
>
> steps, while a classical algorithm needs $O(N \log N)$ steps.

exponential speedup!

period finding and factorization

We now want to discuss how an efficient period finding can be used to determine prime factors of large numbers

given: N what is $k \neq 1, N$ with $N = k \cdot r$
$(N, k, r \in \mathbb{N})$

best classical sieve $O\left[\exp\left((\log N)^{1/3} (\log \log N)^{2/3}\right)\right]$
$\Rightarrow$ "NP" problem (?)

here: make this a "P" problem

(i) choose $a < N$ randomly

(ii) determine largest common divisor of a and N e.g. using Euklid algorithm $\Rightarrow$ poly$(\log N)$
- If such a divisor exists the search is over!
- With much higher probability such a common divisor does not exist $\Rightarrow$ step (iii)

(iii) if a and N have no common divisor:
theorem of Euler:
$\exists$ a number r such that

(85) $$a^r \equiv 1 \mod N$$

(iv) This value r from (85) is period of the function

(86) $$f_{N,a}(x) = a^x \pmod N$$

This period can be found efficiently using QFT with effort $O((\log N)^2)$

(v) If r from step (iv) even $\Rightarrow$

(87) $$\underbrace{(a^{r/2}-1)}_{=\alpha}\underbrace{(a^{r/2}+1)}_{=\beta} = a^r - 1 = 0 \mod N$$

I.e. $\alpha\beta$ is multiple of N

(88) $\alpha\beta = mN$

Determine largest common divisor of (α, N) and (β, N). This can be done efficiently using again the Euler algorithm. Either (α, N) or (β, N) have largest common divisor which is a factor of N!

If r in (iv) is odd $\Rightarrow$ repeat procedure with different a

Determining the integer factors of an integer N can be done using a quantum algorithm in
$\text{poly}(\log(N))$ steps!

$\Rightarrow$ Factorization $\in$ BQP

RSA algorithm of classical cryptography is no longer secure

$\Rightarrow$ use quantum cryptography
or
classical "post-quantum" algorithms that cannot be decoded by a quantum computer