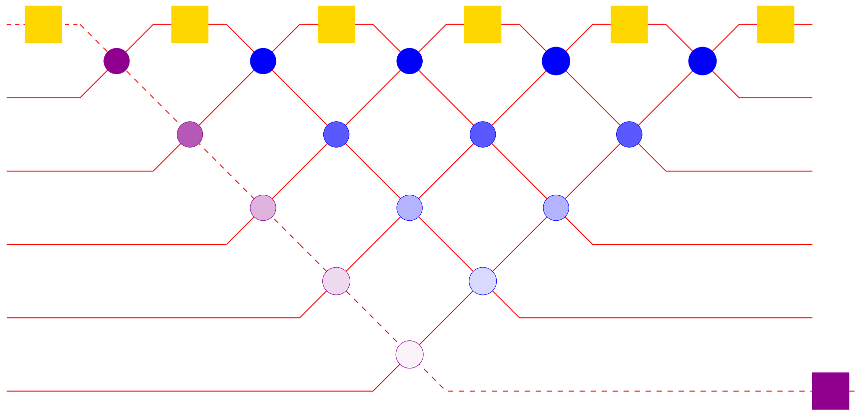
What has quantum mechanics to do with factoring?

Things I wish they had told me about Shor's algorithm



Walter Kohn Lecture

Sherbrooke, 6 November 2007

Question:

What has quantum mechanics to do with factoring?

Question:

What has quantum mechanics to do with factoring?

Answer:

Nothing!

Question:

What has quantum mechanics to do with factoring?

Answer:

Nothing!

But quantum mechanics is good at diagnosing periodicity, which (for purely arithmetic reasons) helps in factoring.

Factoring $N = pq$, where p and q are huge (e.g. 300 digit) primes, follows from ability to find smallest r with $a^r = 1 \pmod N$ for integers a sharing no factors with N.

$a^x \pmod N$ is periodic with period r.

Pick random a. Use quantum computer to find r.

Pray for two pieces of good luck.

Quantum computer gives *least* r with $a^r - 1$ divisible by $N = pq$

First piece of luck: r even.
Then $(a^{r/2} - 1)(a^{r/2} + 1)$ is divisible by N, but $a^{r/2} - 1$ is *not*,

Second piece of luck: $a^{r/2} + 1$ is *also* not divisible by N.
Then product of $a^{r/2} - 1$ and $a^{r/2} + 1$ is divisible by both p and q although neither factor is divisible by both.

Since p, q primes, one factor divisible by p and other divisible by q.
So p is greatest common divisor of N and $a^{r/2} - 1$
and q is greatest common divisor of N and $a^{r/2} + 1$

FINISHED!

Finished, because:

1. Can find greatest common divisor of two integers using simple method known to ancient Greeks: Euclidean algorithm.

2. If a is picked at random, an hour's argument shows that the probability is at least 50% that both pieces of luck will hold.

Amazing (but wrong):

[After the computation] the solutions — the factors of the number being analyzed — will all be in superposition.

— George Johnson, *A Shortcut Through Time.*

[The computer will] try out all the possible factors simultaneously, in superposition, then collapse to reveal the answer.

— *Ibid.*

Unexciting but correct:

A quantum computer is efficient at factoring because it is efficient at period-finding.

Next question: What's so hard about period finding?

Given graph of $\sin(kx)$ it's easy to find the period $2\pi/k$. Since no value repeats inside a period, $a^x \pmod{N}$ is even simpler.

Next question: What's so hard about period finding?

Given graph of $\sin(kx)$ it's easy to find the period $2\pi/k$. Since no value repeats inside a period, $a^x \pmod{N}$ is even simpler.

What makes it hard:

Within a period, unlike the smooth, continuous $\sin(kx)$, the function $a^x \pmod{N}$ looks like random noise.

Nothing in a list of r consecutive values gives a *hint* that the next one will be the same as the first.

PERIOD FINDING WITH A QUANTUM COMPUTER

Represent n bit number

$$x = x_0 + 2x_1 + 4x_2 + \cdots + 2^{n-1}x_{n-1} \qquad \text{(each } x_j \text{ 0 or 1)}$$

by product of states $|0\rangle$ and $|1\rangle$ of n 2-state systems (*Qbits*):

$$|x\rangle = |x_{n-1}\rangle \cdots |x_1\rangle|x_0\rangle$$

↗

Classical or *Computational* basis.

Qbits, not *qubits* because:

1. Classical two state systems are *Cbits* (not *clbits*)
2. Ear cleaners are *Qtips* (not *Qutips*)
3. Dirac wrote about *q-numbers* (not *qunumbers*)

(*q-bit* awkward: *2-Qbit gate* OK;
2-q-bit gate unreadable.)

Represent function f taking n-bit to m-bit integers by a linear, norm-preserving (unitary) transformation $\mathbf{U}_f$ acting on n-Qbit *input register* and m-Qbit *output register*:

input register

$$\downarrow \qquad\qquad \downarrow$$

$$\mathbf{U}_f|x\rangle|0\rangle = |x\rangle|f(x)\rangle.$$

$$\uparrow \qquad\qquad \uparrow$$

output register

(More generally, $\mathbf{U}_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$.

$y \oplus z$ is bitwise modulo 2 sum: $1010 \oplus 0111 = 1101$.)

$$\mathbf{U}_f|x\rangle|0\rangle = |x\rangle|f(x)\rangle$$

Put input register into superposition of all possible inputs:

$$|\phi\rangle = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0\le x<2^n} |x\rangle$$

$$= \tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)\cdots\tfrac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big).$$

Applying linear $\mathbf{U}_f$ gives

$$\mathbf{U}_f\big(|\phi\rangle|0\rangle\big) = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0\le x<2^n} |x\rangle|f(x)\rangle.$$

$$\mathbf{U}_f\big(|\phi\rangle|0\rangle\big) = \big(\tfrac{1}{\sqrt{2}}\big)^n \sum_{0 \le x < 2^n} |x\rangle|f(x)\rangle.$$

Question:

Has *one* invocation of $\mathbf{U}_f$ computed $f(x)$ for *all* x?

$$\mathbf{U}_f\big(|\phi\rangle|0\rangle\big) = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0\le x<2^n} |x\rangle|f(x)\rangle.$$

Question:

Has *one* invocation of $\mathbf{U}_f$ computed $f(x)$ for *all* x?

Answer:

No. Given a single system in an unknown state,
there is no way to learn what that state is.

Information is acquired *only* through measurement.
Direct measurement of input register gives random x_0;
Direct measurement of output register then gives $f(x_0)$.

$$\mathbf{U}_f\big(|\phi\rangle|0\rangle\big) = \big(\tfrac{1}{\sqrt{2}}\big)^n \sum_{0\le x<2^n} |x\rangle|f(x)\rangle.$$

Special form when $f(x) = a^x \pmod N$:

$$\sum_{0\le x<2^n} |x\rangle|a^x\rangle = \sum_{0\le x<r} \Big(|x\rangle + |x+r\rangle + |x+2r\rangle + \cdots\Big)|a^x\rangle$$

Measuring output register leaves input register in state

$$|x\rangle + |x+r\rangle + |x+2r\rangle + \cdots$$

for random $x < r$.

Given n Qbits in the state $|x\rangle + |x+r\rangle + |x+2r\rangle + \cdots$

If you could learn what the state was you would know r.

But there is no way to learn what the state is.

If you could make exact copies of an unknown state
you could learn several random multiples of r.

But there is no way to duplicate an unknown state.

THE QUANTUM FOURIER TRANSFORM (QFT)

$$\mathbf{V}_{FT}|x\rangle = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0 \le y < 2^n} e^{2\pi ixy/2^n}|y\rangle$$

Acting on superpositions, $\mathbf{V}_{FT}$ Fourier-transforms amplitudes:

$$\mathbf{V}_{FT} \sum \alpha(x)|x\rangle = \sum \beta(x)|x\rangle$$

$$\beta(x) = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0 \le z < 2^n} e^{2\pi ixz/2^n}\alpha(z)$$

If α has period r, as in $|x\rangle + |x+r\rangle + |x+2r\rangle + \cdots$, then β is sharply peaked at integral multiples of $2^n/r$.

HO-HUM!

$\mathbf{V}_{FT}$ is *boring*:

1. Just familiar transformation from position to momentum representation.
2. Everybody knows Fourier transform sharply peaked at multiples of inverse period.

But $\mathbf{V}_{FT}$ is *not* ho-humish because:

1. x has nothing to do with position, real or conceptual.

x is arithmetically useful but physically meaningless:

$$x = x_0 + 2x_1 + 4x_2 + 8x_3 + \cdots,$$

where $|x_j\rangle = |0\rangle$ or $|1\rangle$ is state of j-th 2-state system.

2. *Sharp* means sharp compared with resolution of apparatus.

The period r is hundreds of digits long.

Need to know r *exactly* — every single digit.

Error in r of 1 in 10^{10} messes up almost every digit.

$$\mathbf{V}_{FT}\Big(|x\rangle + |x+r\rangle + |x+2r\rangle + \cdots\Big) =$$

$$= \big(\tfrac{1}{\sqrt{2}}\big)^n \sum_{0\le y<2^n} \Big(1 + \alpha + \alpha^2 + \alpha^3 + \cdots\Big) e^{2\pi i x y/2^n}\,|y\rangle,$$

$$\alpha = \exp\Big(2\pi i y/(2^n/r)\Big).$$

Sum of phases α sharply peaked at values of y as close as possible to (i.e. within $\frac{1}{2}$ of) integral multiples of $2^n/r$.

Question: *How sharply peaked?*

Answer: Probability that measurement of input register gives such a value of y exceeds 40%.

Significant ($> 40\%$) chance of learning integer y
as close as possible to (i.e. within $\frac{1}{2}$ of) $j(2^n/r)$
for some (more or less) random integer j.

Then $y/2^n$ is within $1/2^{n+1}$ of j/r.

Question: Does this pin down unique rational number j/r?

Significant ($> 40\%$) chance of getting integer y
as close as possible to (i.e. within $\frac{1}{2}$ of) $j(2^n/r)$
for some (more or less) random integer j.

Then $y/2^n$ is within $1/2^{n+1}$ of j/r.

Question: Does this pin down unique rational number j/r?

Answer: It depends.

Suppose second candidate, j'/r' with $j'/r' \neq j/r$.

$$\left|\frac{j'}{r'} - \frac{j}{r}\right| = \frac{|j'r - jr'|}{rr'} \geq \frac{1}{rr'} \geq \frac{1}{N^2}$$

So answer is Yes, if $2^n > N^2$.

Input register must be large enough to represent N^2.

Then have 40% chance of learning a *divisor* r_0 of r.

(r_0 is r divided by factors it shares with (random) j)

A comment:

When $N = pq$, easy to show period r necessarily $< N/2$. So

$$\left|\frac{j'}{r'} - \frac{j}{r}\right| > \frac{4}{N^2}$$

and therefore don't need y *as close as possible* to integral multiple of $2^n/r$.

Second, third, or fourth closest do just as well.

Raises probability of learning divisor of r from 40% to 90%.

Another comment:

Should the period r be 2^m, then $2^n/r$ is itself an integer, and probability of y being multiple of that integer is easily shown to be 1, even if input register contains just a single period.

A pathologically easy case.

Question: When must all periods r be powers of 2?
Answer: When p and q are both of form $2^j + 1$.
(Periods are divisors of $(p-1)(q-1)$.)

Therefore factoring $15 = (2+1) \times (4+1)$
— i.e. finding periods modulo 15 —
is not a serious demonstration of Shor's algorithm.

Some neat things about the quantum Fourier transform

$$\mathbf{V}_{FT}|x\rangle = \big(\tfrac{1}{\sqrt{2}}\big)^n \sum_{0\le y<2^n} e^{2\pi i x y/2^n}|y\rangle$$

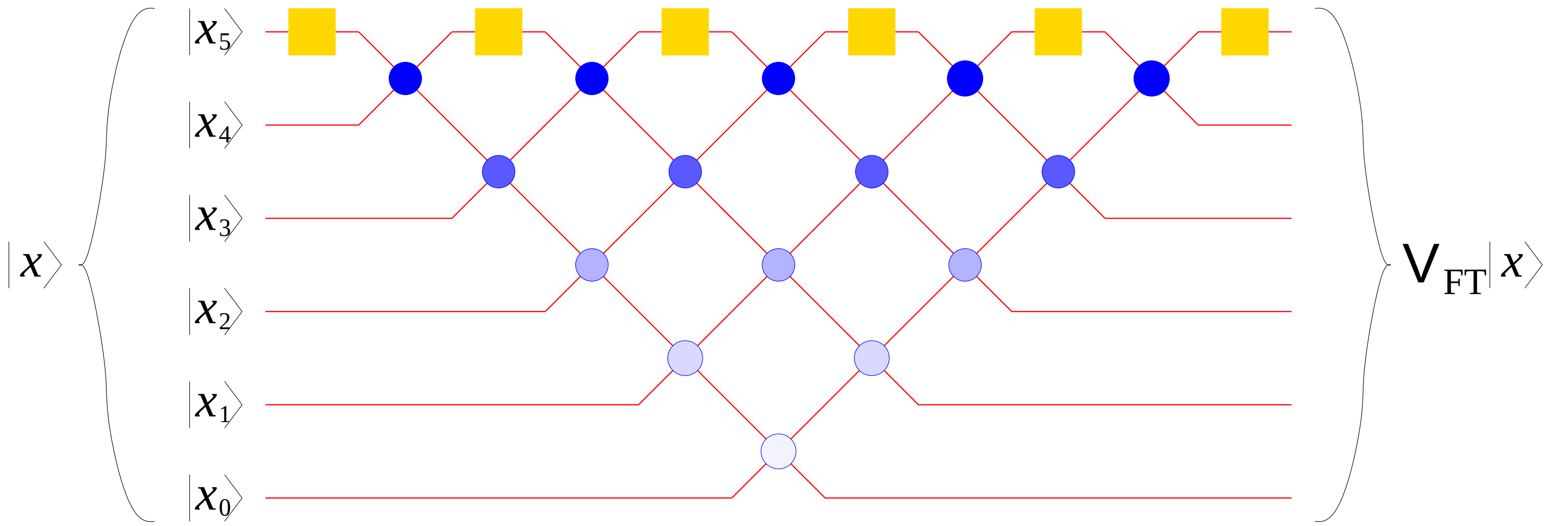
1. Constructed entirely out of 1-Qbit and 2-Qbit gates.
2. Number of gates and therefore time grows only as n^2.
3. With just *one* application,

$$\sum \alpha(x)|x\rangle \longrightarrow \sum \beta(x)|x\rangle,$$

$$\beta(x) = \big(\tfrac{1}{\sqrt{2}}\big)^n \sum_{0\le z<2^n} e^{2\pi i x z/2^n}\alpha(z)$$

In *classical* "Fast Fourier Transform" time grows as $n2^n$.

But classical FFT gives *all* the $\beta(x)$, while QFT gives *only* $\sum \beta(x)|x\rangle$.

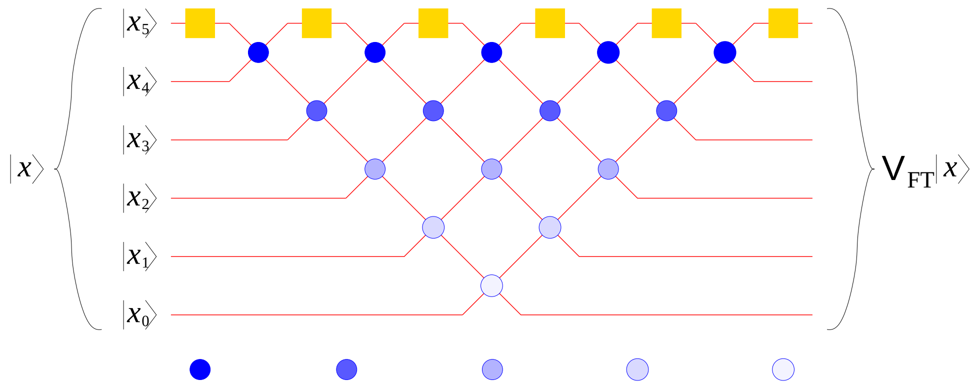


$$\left.\begin{matrix}|0\rangle \\ |1\rangle\end{matrix}\right\} \longrightarrow \left\{\begin{matrix}\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\end{matrix}\right.$$

$e^{\pi i \mathbf{n}\mathbf{n}'/2}$ $e^{\pi i \mathbf{n}\mathbf{n}'/4}$ $e^{\pi i \mathbf{n}\mathbf{n}'/8}$ $e^{\pi i \mathbf{n}\mathbf{n}'/16}$ $e^{\pi i \mathbf{n}\mathbf{n}'/32}$

$$|0\rangle|0\rangle,\ |0\rangle|1\rangle,\ |1\rangle|0\rangle \text{ invariant;} \quad |1\rangle|1\rangle \longrightarrow e^{\pi i/2^j}|1\rangle|1\rangle$$

A PROBLEM?



Number n *of Qbits:* $2^n > N^2$, N *hundreds of digits.*

Phase gates $e^{\pi i \mathbf{nn}'/2^m}$ *impossible to make for most* m,

since can't control strength or time of interactions to better than parts in $10^{10} = 2^{30}$.

But need to learn period r *to parts in* 10^{300} *or more!*

Question:

So is it all based on a silly mistake?

Question:

So is it all based on a silly mistake?

Answer:

No, all is well.

Question:

So is it all based on a silly mistake?

Answer:

No, all is well.

Question:

How can that be?

Question:

So is it all based on a silly mistake?

Answer:

No, all is well.

Question:

How can that be?

Answer:

Because of the quantum-computational interplay between analog and digital.

Quantum Computation is Digital

Information is acquired only by measuring Qbits. The reading of each 1-Qbit measurement gate is only 0 or 1.

The 10^3 *bits of the output* y *of Shor's algorithm are given by the readings (0 or 1) of* 10^3 *1-Qbit measurement gates.*

There is no imprecision in those 10^3 *readings. The output is a definite 300-digit number.*

But is it the number you wanted to learn?

Quantum Computation is Analog

Before a measurement the Qbits are acted on by unitary gates with continuously variable parameters.

These variations affect the amplitudes of the states prior to measurement and therefore they affect the probabilities of the readings of the measurement gates.

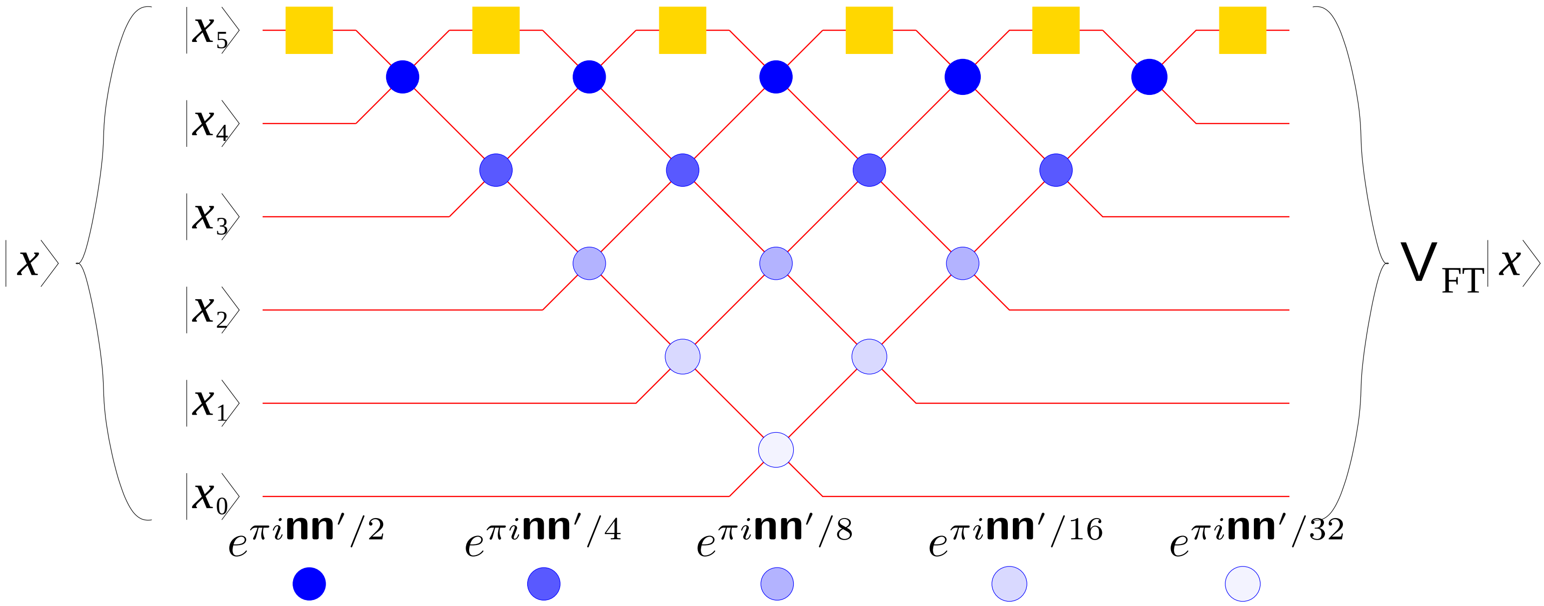
So all is well

"Huge" errors (parts in 10^4) in the phase gates may result in comparable errors in the probability that the 300 digit number given precisely by the measurement gates is the right 300 digit number.

So the probability of getting a useful number may not be 90% but only 89.99%.

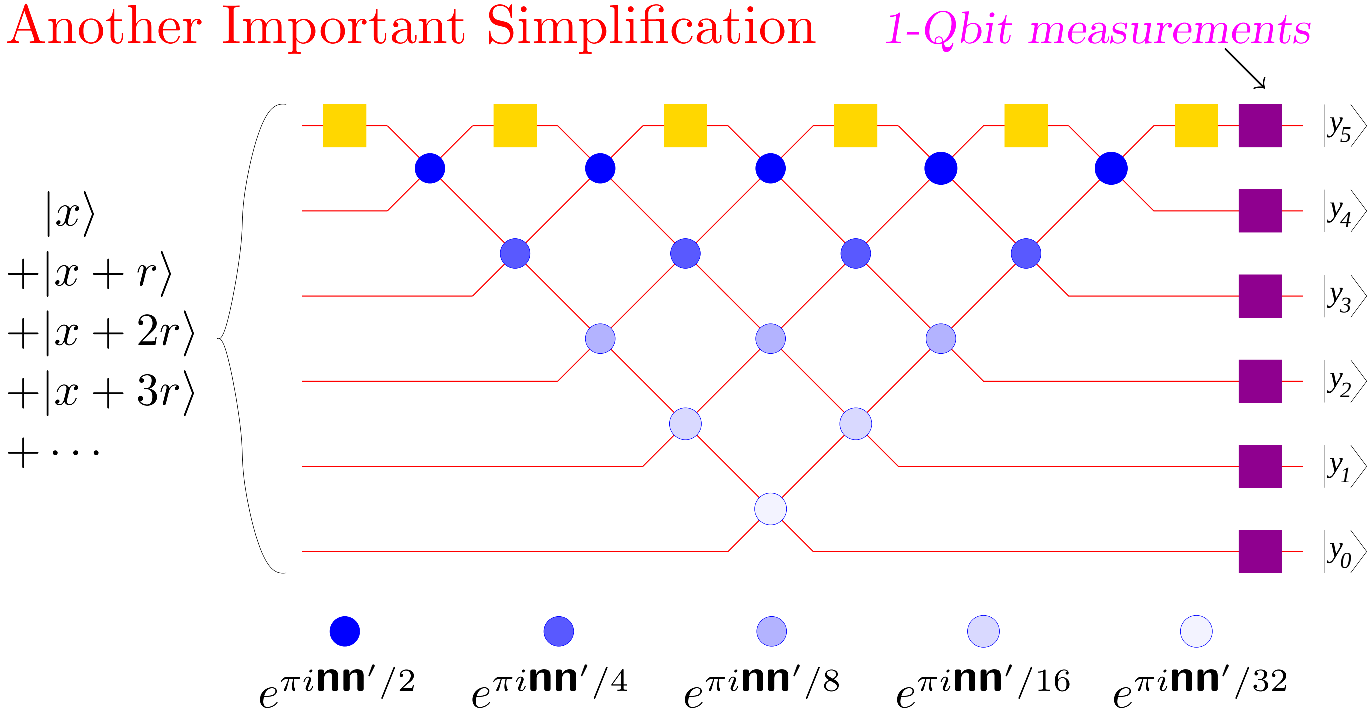
Since "90%" is actually "about 90%" this makes no difference.

In fact this makes things even better

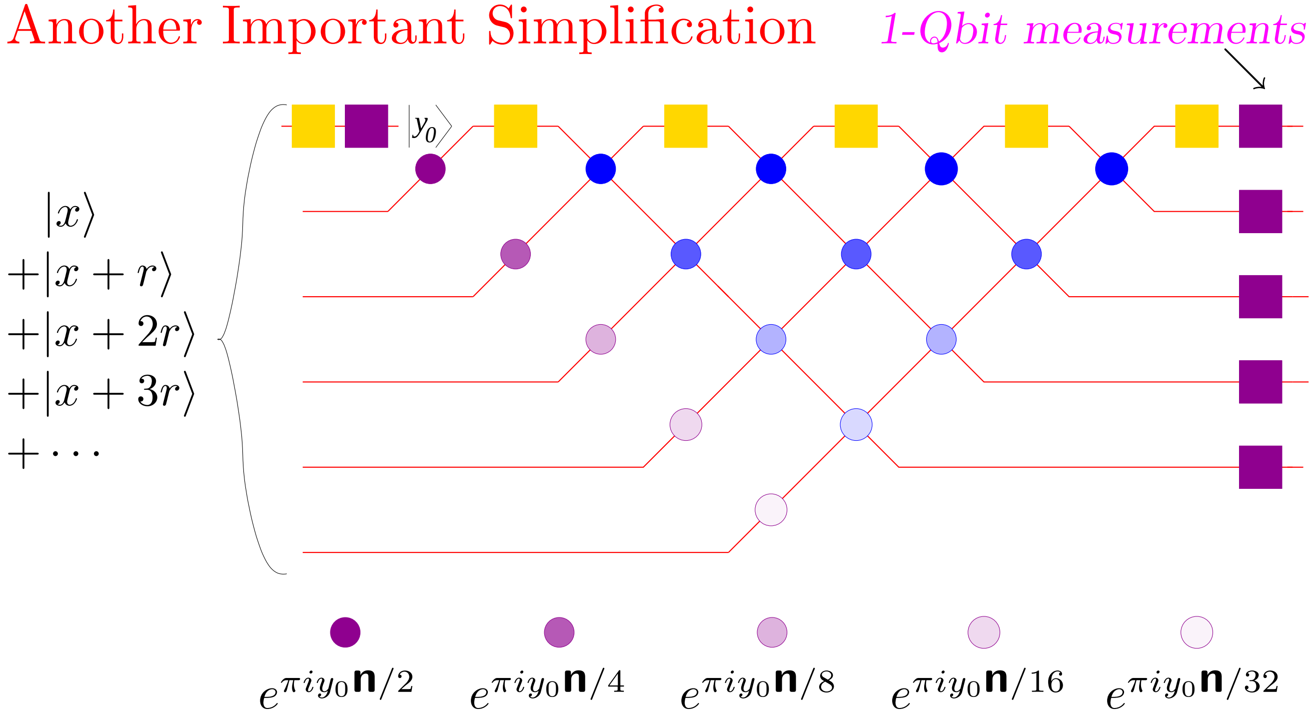


Since only top 20 layers of phase gates matter when $N > 2^{20} = 10^6$,
time for QFT scales not quadratically but linearly in number of Qbits.

Another Important Simplification



Another Important Simplification



You don't need anything but 1-Qbit gates!

Quantum Versus Classical Programming Styles

Question: *How do you calculate* a^x *when* x *is a 300 digit number?*
Answer: *Not by multiplying* a *by itself* 10^{300} *times!*

How else, then?

Write x *as a binary number:* $x = x_{999}x_{998}\cdots x_2x_1x_0$.

Next square a*, square the result, square* that *result* ...,
getting the 1,000 numbers a^{2^j}.

Finally, multiply together all the a^{2^j} *for which* $x_j = 1$.

$$\prod_{j=0}^{999}\left(a^{2^j}\right)^{x_j} = a^{\sum_j x_j 2^j} = a^x$$

Classical: Cbits Cheap; Time Precious

$$a^x = \prod_{j=0}^{999} \left(a^{2^j}\right)^{x_j}$$

Once and for all, make and store a look-up table:

$$a,\ a^2,\ a^4,\ a^8, \ldots,\ a^{2^{999}}$$

A thousand entries, each of a thousand bits.

For each x multiply together all the a^{2^j} in the table for which $x_j = 1$.

Quantum: Time Cheap; Qbits Precious

Circuit that executes

$$a^x = \prod_{j=0}^{999} \left(a^{2^j}\right)^{x_j}$$

is not applied 2^n *times to input register for each* $|x\rangle$*.*
It is applied just once to input register in the state

$$|\phi\rangle = \left(\tfrac{1}{\sqrt{2}}\right)^n \sum_{0\le x<2^n} |x\rangle.$$

So after each conditional (on $x_j = 1$*) multiplication by* a^{2^j}
can store $\left(a^{2^j}\right)^2 = a^{2^{j+1}}$ *using same 1000 Qbits*
that formerly held a^{2^j}*.*

Some other things I wish they had told me:

Question:

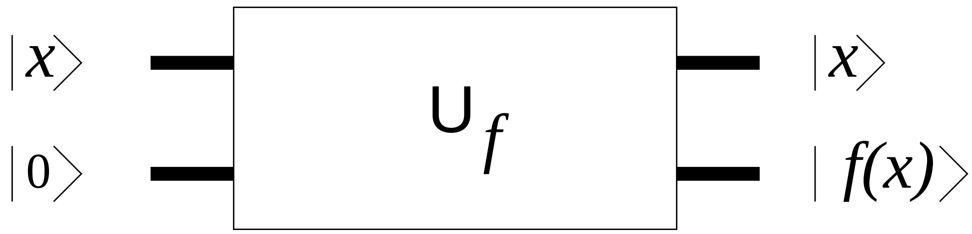
Why must a quantum computation be reversible (except for measurements)?

Superficial answer:

Because linear + norm-preserving ⇒ unitary and unitary transformations have inverses.

Real answer:

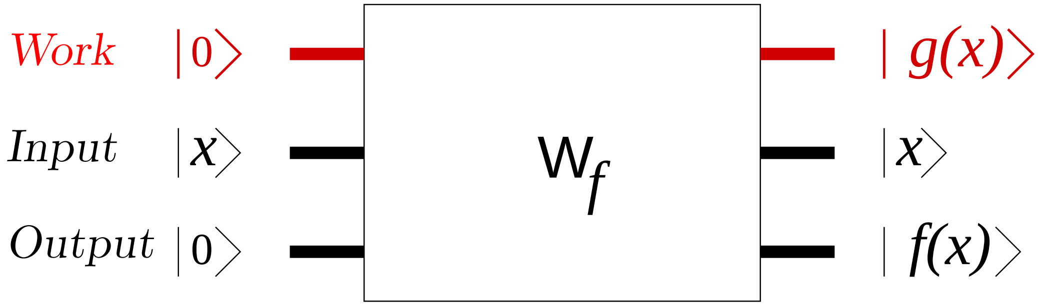
Because standard architecture for evaluating $f(x)$,



oversimplifies *the actual architecture:*

Need additional work registers for doing calculation:

Registers

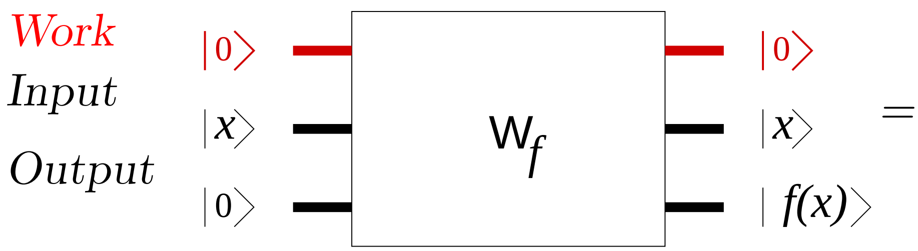


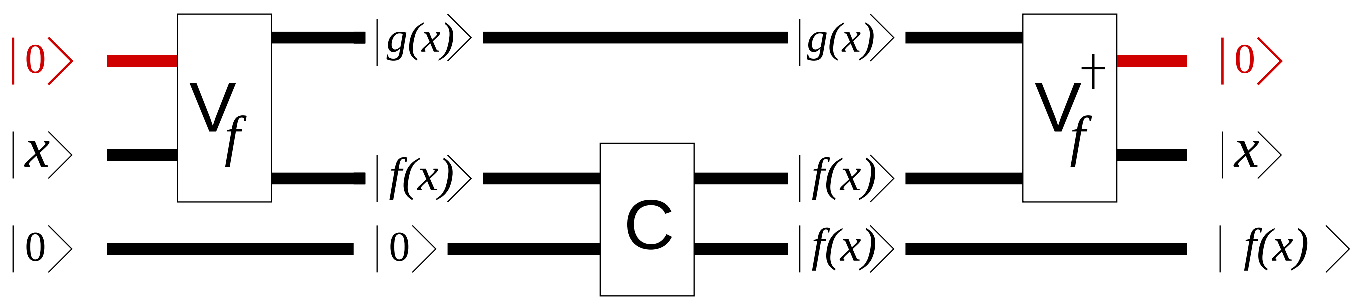
If input register starts in standard state $\sum_x |x\rangle$
then final state of all registers is $\sum_x |g(x)\rangle |x\rangle |f(x)\rangle$.

Work register entangled with input and out registers,
Quantum parallelism breaks down.

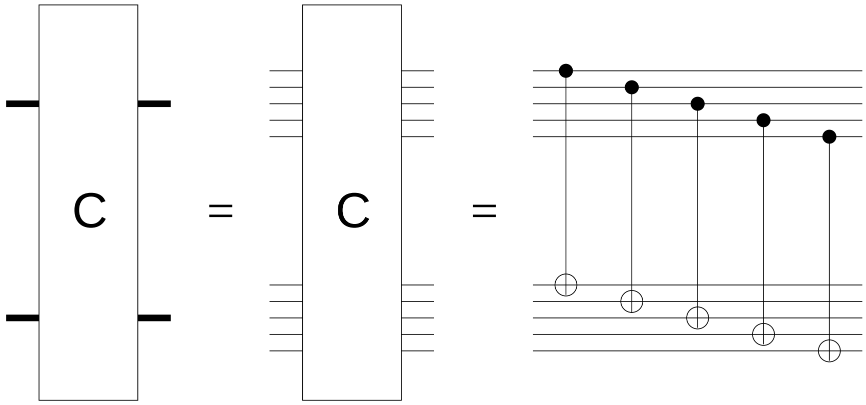
Quantum parallelism maintained if $|g(x)\rangle = |0\rangle$, *for any* x.
Final state is then $|0\rangle \left(\sum_x |x\rangle |f(x)\rangle \right)$.

How to keep the work register unentangled:

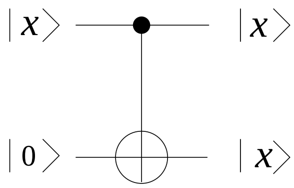




C *is built out of 1-Qbit controlled-NOT gates:*



controlled-NOT:



Question:

How do you do arithmetic on a quantum computer?

Answer:

By copying the (pre-existing) classical theory of reversible computation.

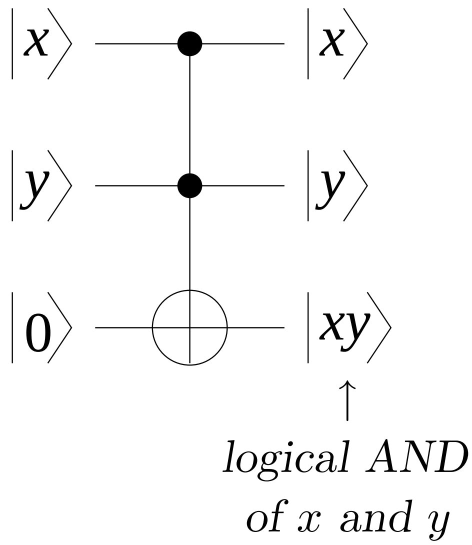
Question (from reversible-classical-computer scientist):

But that theory requires an irreducibly 3-Cbit doubly-controlled-NOT (Toffoli) gate!

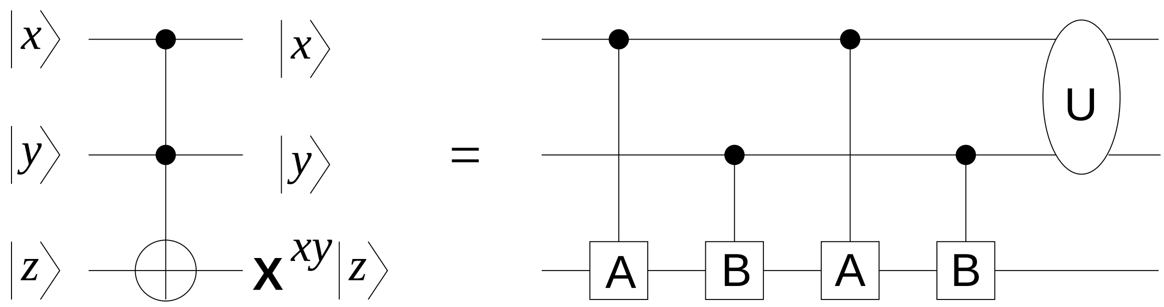
Answer:

In a quantum computer 3-Qbit Toffoli gate can be built from a few 2-Qbit gates.

The 3-Cbit Doubly-Controlled-NOT (Toffoli) gate:



Building 3-Qbit Doubly-Controlled-NOT gate from 2-Qbit gates:



$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \sigma_x \qquad \mathbf{U} = e^{-\pi i \mathbf{n}\mathbf{n}'/2}$$

$$\mathbf{A} = \hat{\mathbf{a}} \cdot \sigma \qquad \mathbf{B} = \hat{\mathbf{b}} \cdot \sigma \qquad \hat{\mathbf{a}} \times \hat{\mathbf{b}} = \hat{\mathbf{x}} \sin\theta \qquad \mathbf{A}^2 = \mathbf{B}^2 = \mathbf{1}$$

$$\mathbf{AB} = \hat{\mathbf{a}} \cdot \hat{\mathbf{b}} + i\hat{\mathbf{a}} \times \hat{\mathbf{b}} \cdot \sigma = \cos\theta + i\sigma_x \sin\theta$$

$$(\mathbf{AB})^2 = \cos 2\theta + i\sigma_x \sin 2\theta$$

If angle θ *between* $\hat{\mathbf{a}}$ *and* $\hat{\mathbf{b}}$ *is* $\pi/4$ *then* $(\mathbf{AB})^2 = i\mathbf{X} = e^{\pi i/2}\mathbf{X}$

Reference:

Quantum Computer Science: An Introduction

N. David Mermin

Cambridge University Press