

Quantum Computing 2

Applications

Applications for Quantum Computing

- Quantum Cryptography
- Quantum Synchronization
- Factoring

Quantum Cryptography

- More accurate name: Quantum Key Distribution
- Idea: One-time pad cryptography has been shown to be “unbreakable”
- Problem: How do you get the pad/key to both participants safely, how do you guarantee randomness
- QKD uses quantum techniques to do this

Quantum Key Distribution

- Generate stream of EPR pairs, give one to sender and one to receiver
- Sender and receiver measure their particles for spin orientation at an angle to the original system -- generates stream of random bits
- Sender and receiver compare some of the bits they got to see if they match. If so, assume nobody has observed the stream

Trying to Break QKD

- One option is to only observe some of the bits, but the fraction of the information you get goes down significantly if you want a high chance of being unobserved
- Other approach: gain control of the particle generation process, and actually generate triplets of entangled particles. Paper talks about ways you can detect this.

Quantum Clock Synchronization

- Problem: Determine the offset between two clocks in physically-separate areas. (assume that the clocks keep perfect time)
- Complication: Time to communicate between the two clocks may be variable, and this variation introduces limits in the synchronization accuracy

Idea

- Many quantum particles have a phase that changes at some rate (ω), making them “timepieces”
- Can use this to obtain accurate measurements of time difference

Procedure

- Alice sends Bob three things: the time at her location (t_{a1}), a particle (ψ), and w , the rate at which the particle's phase changes.
- When Bob receives the particle (at time t_{b2}), it has evolved to state $e^{iw^*(\text{transit time})} Z$ times its original state.
- Bob transforms the particle by $X e^{-(iw(t_{b2}-t_{a1}))Z}$, giving him $X e^{-iwZ(\Delta t)} (\psi)$

Procedure 2

- Bob sends this particle back to Alice, who does a similar transformation to get $e^{-2iwZ(\text{delta})}(\text{psi})$
- If Alice observes this particle, it will be in state 0 with probability $\cos^2(2w(\text{delta}))$. By making many observations of particles that undergo this process, she can obtain delta to arbitrary accuracy.
- Paper shows how to reduce # of observ.

Shor's Factoring Algorithm

- First “killer app” for quantum computing

Factoring numbers on conventional computers

Best known algorithm basically involves trying all the possibilities

Exponential growth in execution time with length of the number being factored

No proof that better algorithm doesn't exist for conventional computers

Definitions and previous math

A number x has order r (using mod- n computations) if $x^r = 1 \pmod{n}$, and r is the smallest integer for which this is true.

Given a number n , a random number x , and $r = \text{order}(x) \pmod{n}$, there is a known algorithm that has at least a 50% chance of finding non-trivial factors of n in polynomial time.

Depends on x , can iterate with different values of x

Shor's contribution: quantum algorithm that finds r in polynomial time given x and n

Factoring Algorithm

- Machine requires two quantum registers, and temporary state
- Given X and N , first find L , the power of two between N^2 and $2N^2$
- Place register 1 in superposition of all possible states representing numbers $a \pmod{L}$
- Compute register 2 = $X^b \pmod{N}$

Factoring Algorithm

- Perform quantum Fourier transform of the first register
- Observe the machine. It is now in a state where the probability peaks of the system are such that, with high probability, you can find r , which can be easily checked
- May have to iterate a few times to get r , and to get a value of r that gives a useful factorization.

