Quantum Computers: The End of Cryptography?

By Andre Infante

Read the original article here: http://www.makeuseof.com/tag/quantum-computers-end-cryptography/

Quantum computing is one of those technologies that's so arcane that TV characters name drop it when they want to sound smart.

Quantum computing as an idea has been around for a while — the theoretical possibility was originally introduced by Yuri Manin and Richard Feynman in 1982. Over the last few years, though, the field has been edging worryingly closer to practicality.

Companies like Google and Microsoft, as well as government agencies like the NSA have all been feverishly pursuing quantum computers for years now. A company called D-Wave has produced and is selling devices that (while they aren't proper computers, and can only perform a few algorithms) exploit quantum properties, and are another incremental step on the road toward a fully Turing-complete quantum machine.

It doesn't seem unreasonable to say that breakthroughs might occur that will allow the first large-scale quantum computer to be built within a decade.

So why all the interest? Why should you care? Computers get faster all the time – what's so special about quantum computers?

In order to explain why these machines are so important, we're going to have to take a step back and explore exactly what quantum computers are, and why they work. To start off, let's talk about a concept called "runtime complexity."

What is Runtime Complexity?

One of the big surprises in the early days of computer science was the discovery that, if you have a computer that solves a problem of a certain size in a certain amount of time, doubling the speed of the computer does not necessarily let it tackle problems twice as big.

Some algorithms increase in total execution time very, very quickly as the size of the problem grows – some algorithms can be rapidly completed given 100 data points, but completing the algorithm given 1000 data points would require a computer the size of the Earth running for a billion years. Runtime complexity is a formalization of this idea: it looks at the curve of how fast the complexity of a problem grows, and uses the shape of that curve to classify the algorithm.

Generally, these classes of difficulty are expressed as functions. An algorithm that gets proportionately harder when the data set its working on increases (like a simple counting function) is said to be a function with a runtime complexity of "**n**" (as in, it takes **n** units of time to process **n** data points).

Alternately, it might be called "linear", because when you graph it, you get a straight line. Other functions might be **n^2** or **2^n** or **n!** (n factorial). These are polynomial and exponential. In the latter two cases, the exponential ones grow so quickly that in almost all cases they can't be solved for anything except very trivial examples.

Runtime Complexity and Cryptography

If you're hearing this stuff for the first time and it sounds meaningless and arcane, let's try to ground this discussion. Runtime complexity is critical for cryptography, which relies on making decryption much easier for people who know a secret key than for those who don't. In an ideal cryptographic scheme, decryption should be linear if you have the key, and **2^k** (where k is the number of bits in the key) if you don't.

In other words, the best algorithm for decrypting the message without the key ought to be simply guessing possible keys, which is intractable for keys only a few hundred bits long.

For symmetric key cryptography (in which the two parties have the chance to exchange a secret securely before they start communication) this is pretty easy. For asymmetric cryptography, it's harder.

Asymmetric cryptography, in which the encryption and decryption keys are different and can't be easily computed from one another, is a much harder mathematical structure to implement than symmetric cryptography, but it's also a lot more powerful: asymmetric crypto lets you have private conversations, even over tapped lines! It also allows you to create "digital signatures" to allow you to verify who a message came from, and that it hasn't been tampered with.

These are powerful tools, and make up the foundation of modern privacy: without asymmetric cryptography, users of electronic devices would have no reliable protection against prying eyes.

Because asymmetric cryptography is harder to build than symmetric, the standard encryption schemes that are in use today are not as strong as they could be: the most common encryption standard, RSA, can be cracked if you can efficiently find the prime factors of a very large number. The good news is that that's a very hard problem.

The best known algorithm for factoring large numbers into their component primes is called the general number field sieve, and has a runtime complexity that grows a little slower than **2^n**. As a consequence, keys have to be about ten times longer in order to provide similar security, which is something that people normally tolerate as a cost of doing business. The bad news is that the entire playing field changes when quantum computers get thrown into the mix.

Quantum Computers: Changing the Crypto Game

Quantum computers work because they can have multiple internal states at the same time, through a quantum phenomenon called "superposition". That means that they can attack different parts of a problem simultaneously, split across possible versions of the universe. They can also be configured such that the branches that solve the problem wind up with the most amplitude, so that when you open the box on Schrodinger's cat, the version of the internal state that you're most likely to be presented with is a smug-looking cat holding a decrypted message.

For more information about quantum computers, check out our recent article on the subject!

The upshot of this is that quantum computers aren't just linearly faster, the way normal computers are: getting two or ten or a hundred times faster doesn't help much when it comes to conventional cryptography that you're hundreds of billions of times too slow to process. Quantum computers support algorithms that have smaller-growing run time complexities than are otherwise possible. This is what makes quantum computers fundamentally different from other future computational technologies, like graphene and memrister computation.

For a concrete example, Shor's Algorithm, which can only be executed on a quantum computer, can factor large numbers in **log(n)^3** time, which is drastically better than the best classical attack. Using the general number field sieve to factor a number with 2048 bits takes about 10^41 units of time, which works out to more than a trillion trillion trillion. Using Shor's algorithm, the same problem only takes about 1000 units of time.

The effect gets more pronounced the longer the keys are. That's the power of quantum computers.

Don't get me wrong – quantum computers have a lot of potential non-evil uses. Quantum computers can efficiently solve the travelling salesman problem, allowing researchers to build more efficient shipping networks and design better circuits. Quantum computers already have powerful uses in artificial intelligence.

That said, their role in cryptography is going to be catastrophic. The encryption technologies that allow our world to keep functioning depend on the integer factorization problem being hard to solve. RSA and related encryption schemes are what let you trust you're on the right website, that the files you download aren't riddled with malware, and that people aren't spying on your Internet browsing (if you're using Tor).

Cryptography keeps your bank account safe and secures the world's nuclear infrastructure. When quantum computers become practical, all of that technology stops working. The first organization to develop a quantum computer, if the world still works on the technologies we use today, is going to be in a frighteningly powerful position.

So, is the quantum apocalypse inevitable? Is there anything we can do about it? As it turns out… yes.

Post-Quantum Cryptography

There are several classes of encryption algorithms that, as far as we know, are not significantly faster to solve on a quantum computer. These are known collectively as post-quantum cryptography, and provide some hope that the world can transition to cryptosystems that will remain secure in a world of quantum encryption.

Promising candidates include lattice-based encryption, like Ring-Learning With Error, which derives its security from a demonstrably complex machine learning problem, and multivariate cryptography, which derives its security from the difficulty of solving very large systems of simple equations. You can read more about this topic on the Wikipedia article. Beware: a lot of this stuff is complex, and you may find that your mathematics background needs to be beefed up considerably before you can really dig into the details.

The takeaway from a lot of this is that post-quantum cryptoschemes are very cool, but also very young. They need more work to be efficient and practical, and also to demonstrate that they are secure. The reason that we are able to trust cryptosystems is because we've thrown enough clinically paranoid geniuses at them for long enough that any obvious shortcomings would have been discovered by now, and researchers have proved various characteristics that make them strong.

Modern cryptography depends on light as a disinfectant, and most of the post-quantum cryptographic schemes are simply too new to trust world security to. They're getting there, though, and with a little luck and some preparation, security experts can complete the switch before the first quantum computer ever comes on line.

If they fail, however, the consequences may be dire. The thought of anyone having that kind of power is unsettling, even if you're optimistic about their intentions. The question of who first develops a working quantum computer is one that everyone should watch very carefully as we move into the next decade.

Are you concerned about the insecurity of cryptography to quantum computers? What's your take? Share your thoughts in the comments below!

Image Credits: Binary orb Via Shutterstock