

**Some topics
adjacent to
algorithms**

1. Quantum Computing

Idea: Harness the quantum nature of the universe to achieve faster computation.

Classical physics: A bit can be either 0 or 1, or randomly chosen from a distribution on 0,1.

Quantum physics: A bit can be in a superposition state like

$$a_0 \cdot |0\rangle + a_1 \cdot |1\rangle,$$

Where a_0, a_1 are complex numbers with $|a_0|^2 + |a_1|^2 = 1$.

Quantum physics

A bit can be in a superposition state like $a_0 \cdot |0\rangle + a_1 \cdot |1\rangle$, with a_0, a_1 complex numbers such that $|a_0|^2 + |a_1|^2 = 1$.

The bit can be *measured*. The outcome is:

$$\Pr[\text{bit} = b] = |a_b|^2.$$

More generally, a quantum state on n bits is

$$\sum_{x \in \{0,1\}^n} a_x \cdot |x\rangle, \text{ with } \sum_{x \in \{0,1\}^n} |a_x|^2 = 1.$$

If we *measure* the first bit, the outcome is $\Pr[x_1 = b]$ is $\sum_{x_1=b} |a_x|^2$.

Quantum Computing

More generally, a quantum state on n bits is

$$\sum_{x \in \{0,1\}^n} a_x \cdot |x\rangle, \text{ with } \sum_{x \in \{0,1\}^n} |a_x|^2 = 1.$$

Each *quantum computation* step is allowed to apply a “unitary operator” (basically a rotation) to two of the qbits. This induces a rotation of the entire vector in the natural way.

Quantum algorithm: sequence of such simple unitary operators on pairs of qbits + measurement.

Prototypical algorithm: Schor’s algorithm for factoring. Factors numbers in polynomial time, something we do not know how to do with classical algorithms.

Common misconceptions

1. A quantum computer searches through exponentially many possibilities at once.
2. A quantum computer would prove $P=NP$.
3. A quantum computer would speed up *many* algorithms.
4. We have built a quantum computer.

2. Cryptography

Idea: Harness the fact that algorithmic tasks are difficult to achieve secrecy, privacy ...

*Prototypical example: **RSA encryption***

1. User picks $n = pq$, with p, q prime, and uses this data to pick e, d .
Public key = (n, e) .
2. To send message m to user, send $m^e \bmod n$.
3. To decrypt message message compute $(m^e)^d = m \bmod n$.

Claim: Any method that can be used to break this can be used to factor $n = pq$.

2. Cryptography

Notes:

1. Based on hardness of factoring, discrete log etc
2. Used everywhere by every device.
3. Much of it is provably secure only under assumptions: in particular if $P = NP$, most crypto systems can be hacked in polynomial time.

3. Distributed computing

Idea: n processors are in a distributed environment. Some of them are faulty (will not run algorithm correctly, may even be adversarial). Processors exchange messages.

*Prototypical example: **Byzantine agreement***

1. All communication is over private channels.
2. One of the processors A wants to send a bit b to all others.
3. If A is not faulty, all unfaulty processors should output b .
4. Whether or not A is faulty, all unfaulty processors should output same value.

Solution: There is a protocol achieving this number of faulty processors is at most t , and total number of processors is at least $3t + 1$.