

## Lecture 2

### Problem significance.

1. SATISFIABILITY: very general problem that models any problem that you want to solve brute force by trying all possible solutions.
2. Factoring: online security assumes this problem is hard to solve. More discussion: opposite of multiplying which is easy, which is why it's suitable for cryptography: encryption uses multiplication, decryption uses factoring.  
RSA public-key encryption: merchant: choose  $p, q$  large distinct prime numbers,  $x$  coprime to  $(p-1)(q-1)$  and compute  $y$  such that  $xy = 1 \pmod{(p-1)(q-1)}$ ; sk =  $(N, x)$ , pk  $(N, y)$ . customer: encrypt  $m \in \mathbb{Z}_N$  into ciphertext  $c := m^y$ . merchant: decrypt by  $c^x$ . Claim  $c^x = m \pmod N$ . Proof by Fermat's little theorem.
3. Simulating quantum systems: discovering new drugs, batteries, and catalysts. A catalyst that has received a lot of attention: FeMoCo: could help with converting nitrogen + hydrogen into ammonia at normal pressures/temperatures. (Currently done using Haber-Bosch process which is very high-pressure and high-temperature.)

**A remark on randomized computation.** It is important to distinguish between when a problem's speedup is due to randomness vs due to quantumness. Consider the following two problems:

1. given a string of  $n$  bits that's either all zeros or half zero and half one but you don't know where they're placed: decide which is the case. Randomized  $O(1)$ , deterministic  $\Omega(n)$ .
2. NAND tree on  $n := 2^h$  variables. Randomized: consider a randomized algorithm that examines the left or right branch uniformly at random. If it computes 0 in one branch, it just outputs 1 – this is okay by property of NAND. Let  $\alpha_b(h)$  be the maximum complexity of this algorithm when run on inputs that map to  $b$ . Then

$$\alpha_0(h) \leq 2\alpha_1(h-1), \quad (4)$$

$$\alpha_1(h) \leq \alpha_0(h-1) + \alpha_1(h-1)/2. \quad (5)$$

Solves to

$$\left((1 + \sqrt{33})/4\right)^h = O(n^{0.754}), \quad (6)$$

in fact this is the optimal randomized complexity – see Saks and Wigderson '86. Deterministic: can show it's  $\Omega(n)$ .

A considerable part of this class will study randomized computation for two reasons:

1. quantum computation can be used to perform randomized computation: in fact, many quantum algorithms, such as Shor's, have some non-trivial randomized (but non-quantum) component.
2. we want to show quantum computers can be strictly faster than randomized computer, so we need to consider the limits of the latter.

### Timeline.

- Today: 100s qubits, 1000s of operations. A single operation acts between two qubits (think of it as the generalization of a Boolean logic gate like AND). Operations limited by qubit interacting with environment and losing its quantum state – “decoherence”.
- Companies target: 100,000 - 1 million qubits by 2030 (first number IBM, second number Google). Not as crazy as it sounds if a “quantum Moore's law” holds (maybe holds? see chart but you can be the judge.)
- Context:
  1. Breaking online security (factoring RSA-2048): 20 million qubits, 3 billion operations. Gidney and Ekerå, Quantum 2021
  2. Simulating FeMoco: 4 million qubits, 5 billion operations. Lee,...,Babbush, PRX Quantum 2021

### Postulates of quantum information

As a concrete motivation: discussion of the CHSH game.