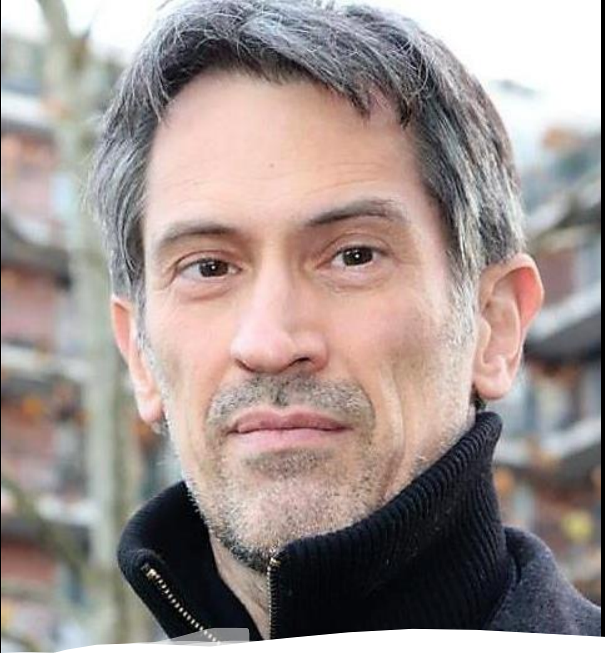
Quantum Computing for Information Retrieval and Recommender Systems




Paolo Cremonesi

Maurizio Ferrari Dacrema

Nicola Ferro

Andrea Pasin

Politecnico di Milano

Università di Padova

Useful References

- Ferrari Dacrema, Moroni, Nembrini, Ferro, Faggioli, Cremonesi.
  Towards feature selection for ranking and classification exploiting quantum annealers.
  SIGIR 2022
  https://doi.org/10.1145/3477495.3531755
- Nembrini, Carugno, Ferrari Dacrema, Cremonesi.
  Towards recommender systems with community detection and quantum computing.
  RecSys 2022
  https://doi.org/10.1145/3523227.3551478
- Nembrini, Ferrari Dacrema, Cremonesi.
  Feature selection for recommender systems with quantum computing.
  Entropy 2021
  https://doi.org/10.3390/e23080970

Outline of the Tutorial

- Part 1: QC Foundations (40 min, Paolo)
  - Introduction to QC
  - Introduction to QA
- Part 2: QUBO Formulation (50 min, Maurizio)
  - How to write NP-complete binary decision problems in QUBO formulation
  - Feature selection and clustering with QA
  - Architecture of a Quantum Annealer: number of available qubits and their topology
- Break
- Part 3: Evaluation of QC for IR and RS (20 min, Nicola)
  - Effectiveness and efficiency
  - The QuantumCLEF lab
- Part 4: Hands-on (70 min, Andrea)
  - The QuantumCLEF infrastructure
  - How to program a Quantum Annealer
  - Hands-on: feature selection and clustering

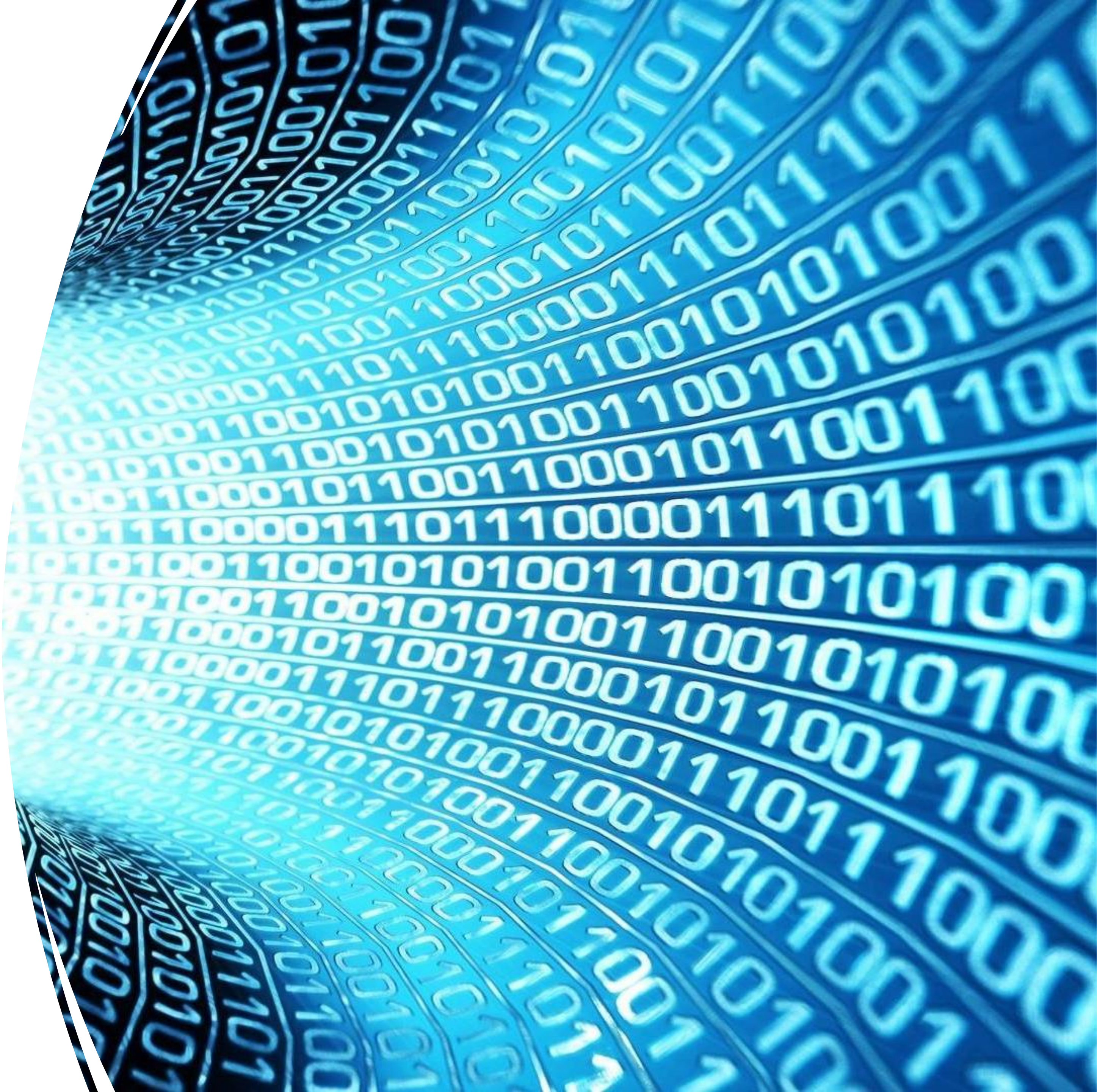
Part 1.

Quantum Computing Foundations

(High Level)
Introduction to Quantum Computing

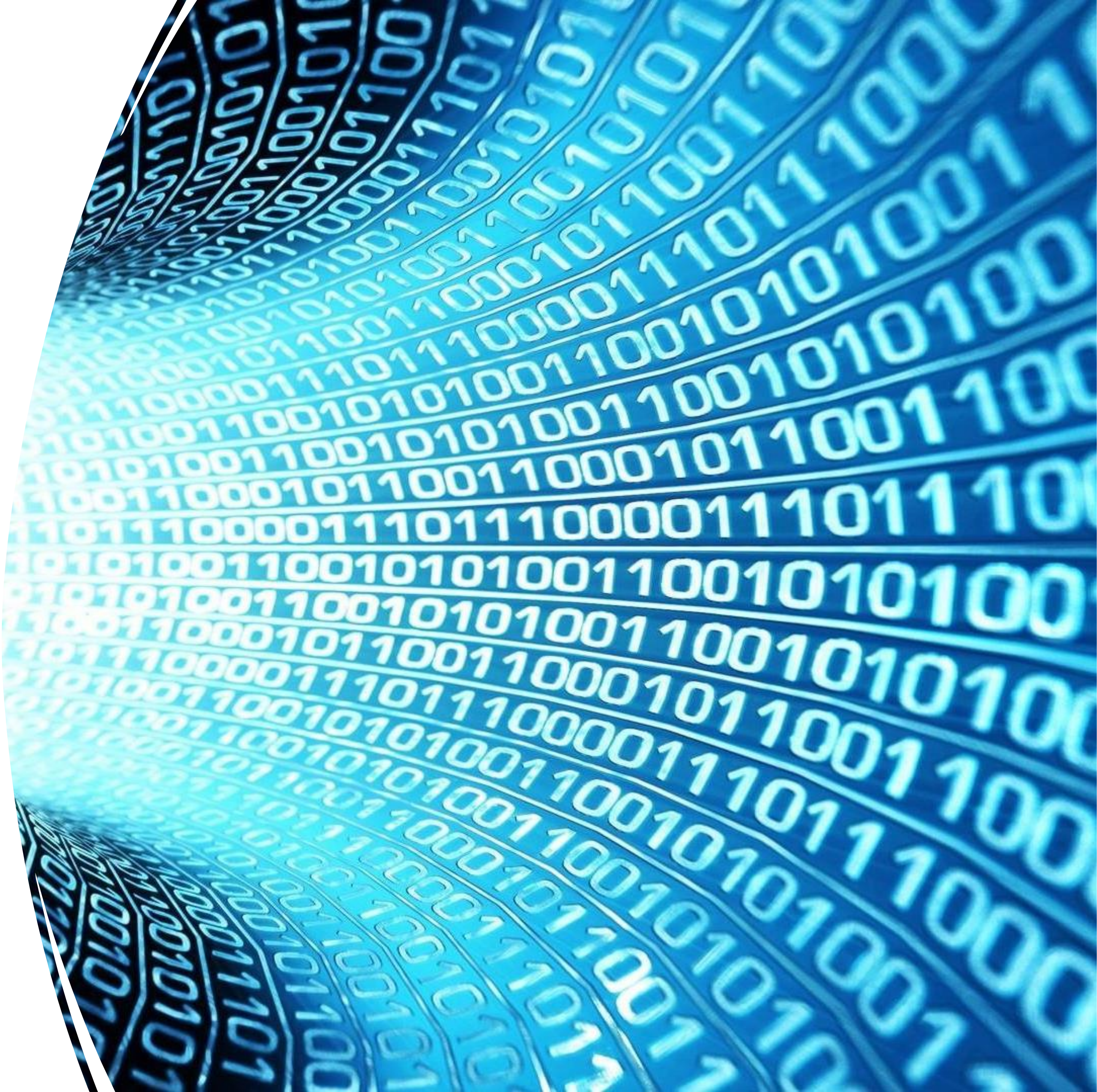
A quantum computer ...

- .... It's not just a more powerful version of the computers we use today

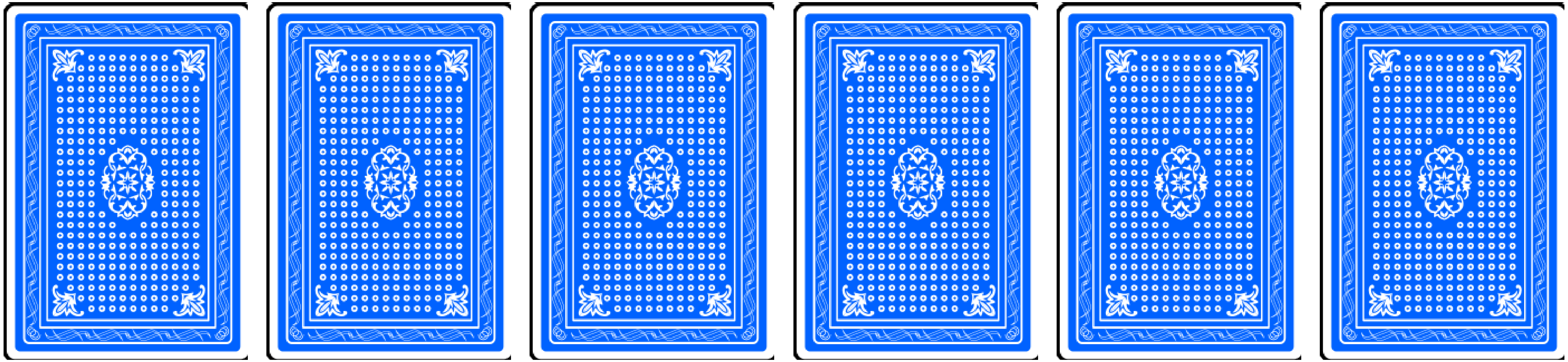


A quantum computer ...

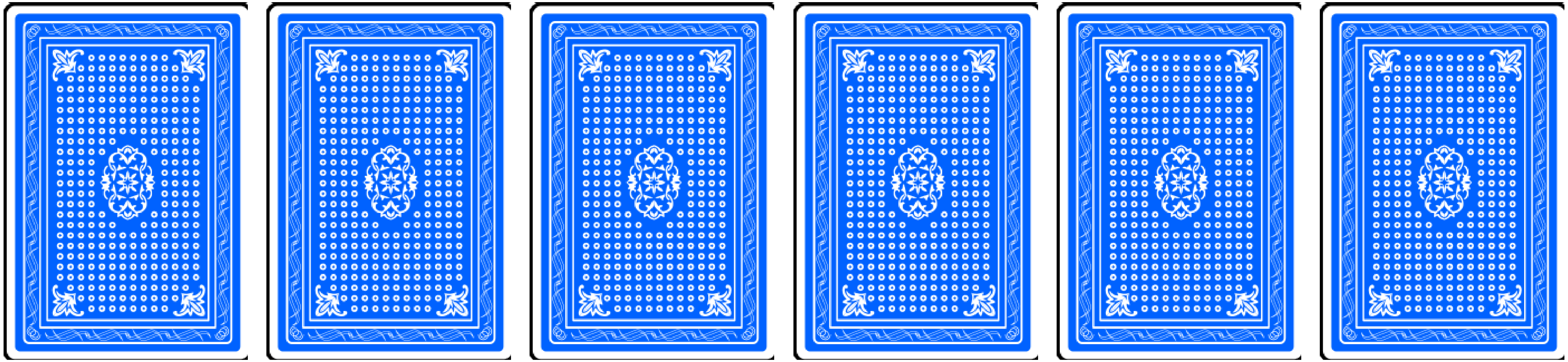
- ... It is something completely different, based on new and **seemingly mysterious** scientific knowledge, where the boundary between reality and science fiction is blurring



Play cards with a QC: Deutsch–Jozsa algorithm

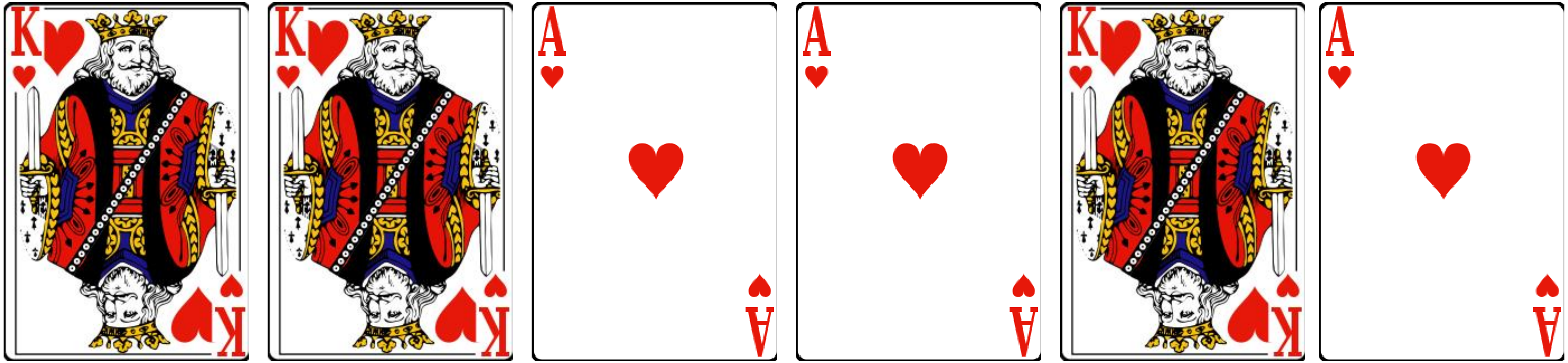


Play cards with a QC: Deutsch–Jozsa algorithm



In this game there are three scenarios

Play cards with a QC: Deutsch–Jozsa algorithm

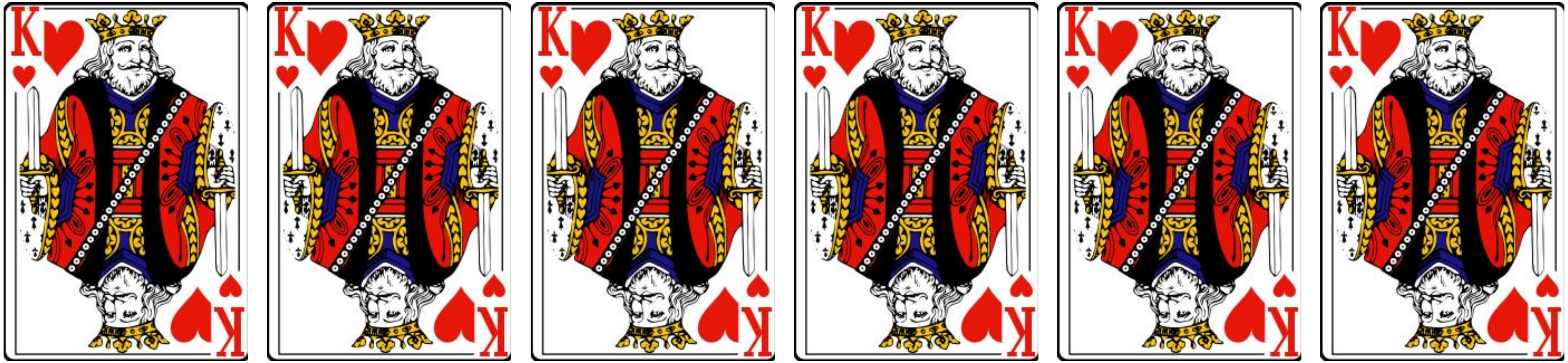


Scenario 1: Balanced

50% Kings

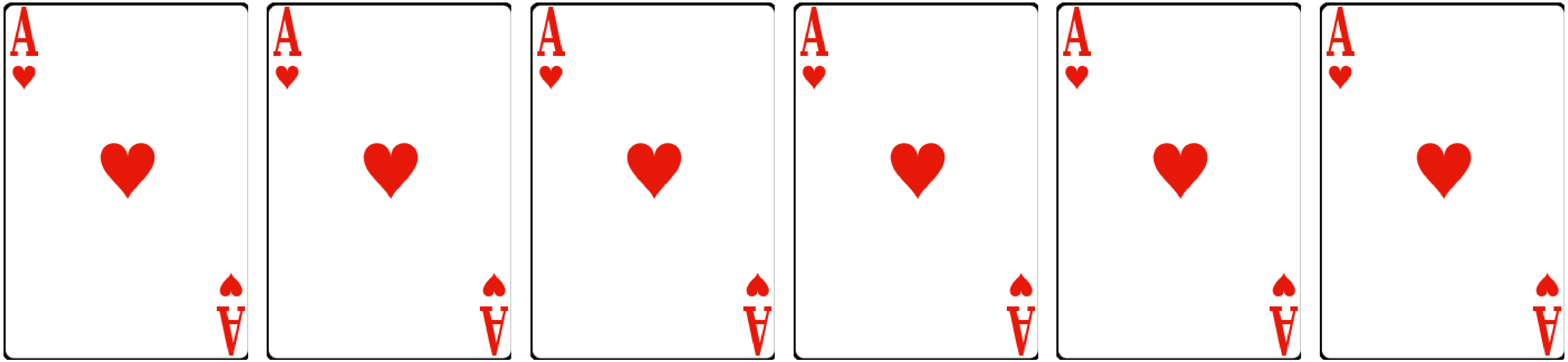
50% Aces

Play cards with a QC: Deutsch–Jozsa algorithm



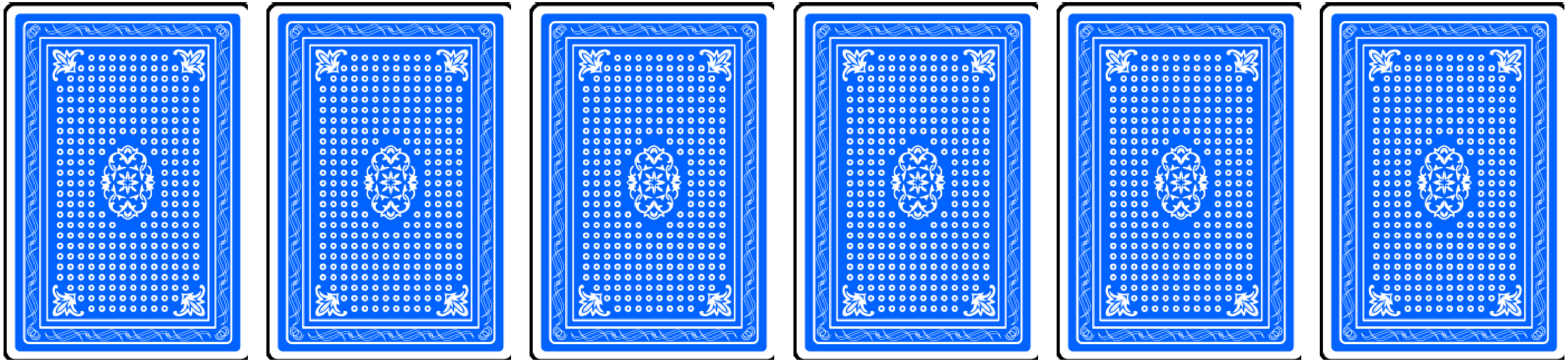
Scenario 2: All Kings

Play cards with a QC: Deutsch–Jozsa algorithm



Scenario 3: All Kings

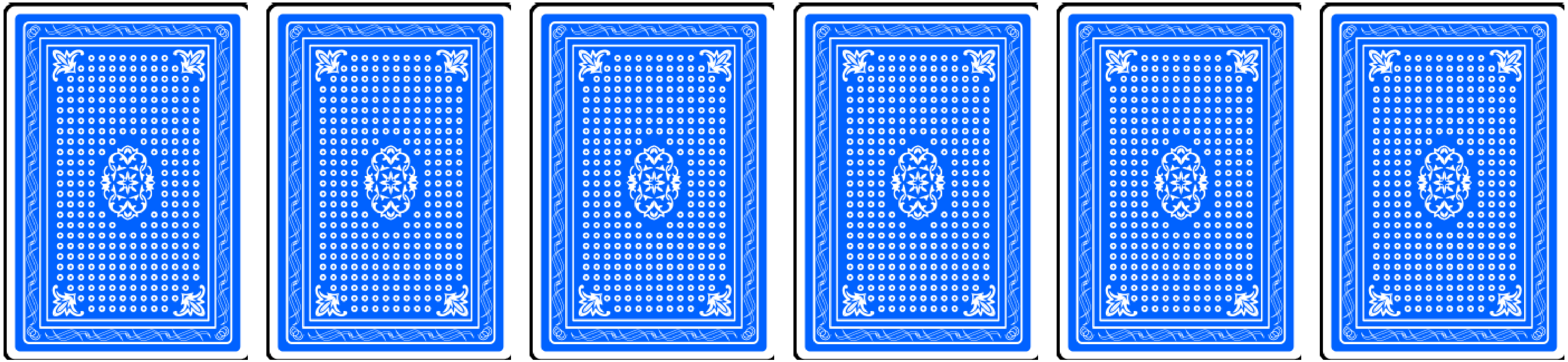
Play cards with a QC: Deutsch–Jozsa algorithm



Cards are now covered and shuffled

How many cards do we need to look at to discover if we are in scenario 1, 2 or 3?

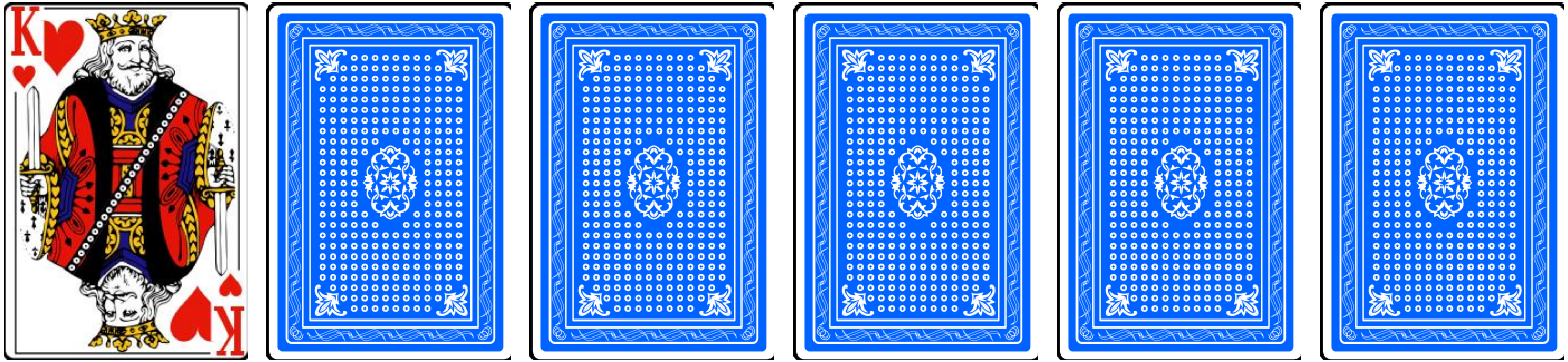
Play cards with a QC: Deutsch–Jozsa algorithm



Classical Computing:

$n/2 + 1$

Play cards with a QC: Deutsch–Jozsa algorithm



Quantum Computing:

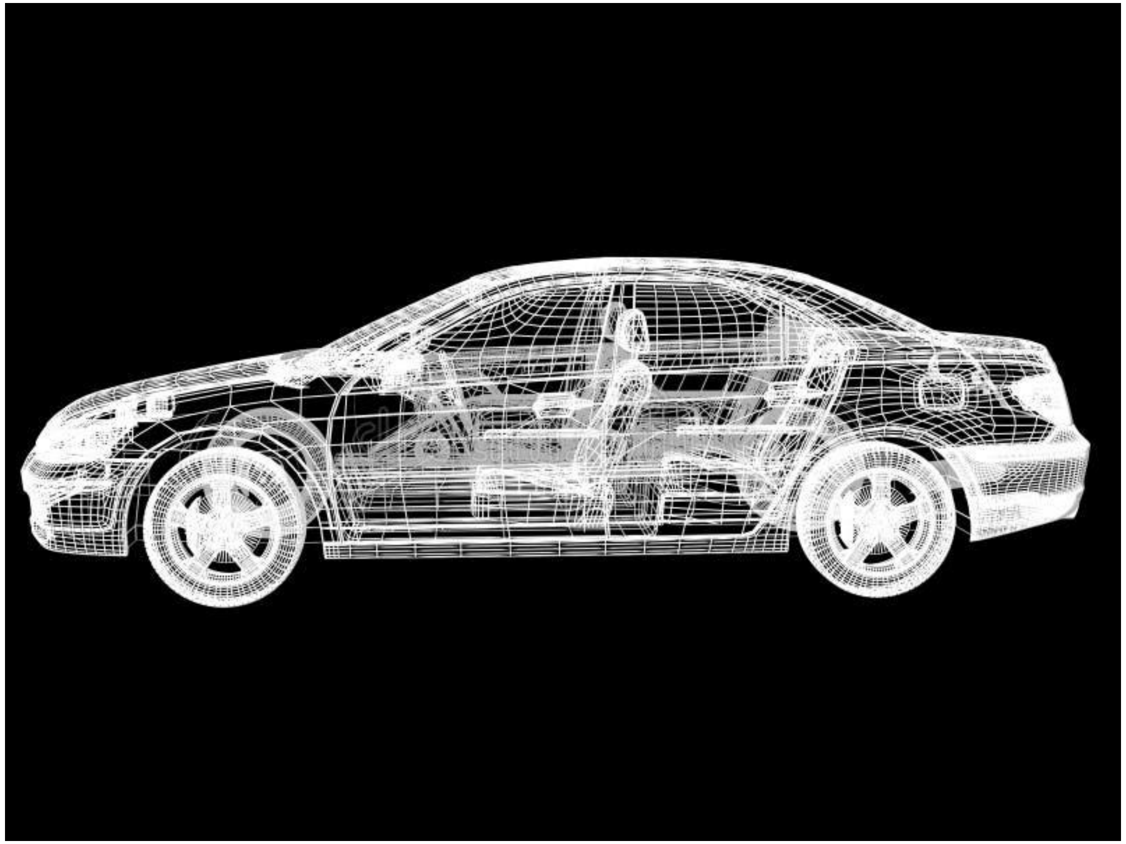
only one !

Build a quantum computer

In the world there are **normal** and **quantum** objects

In the world there are **normal** and **quantum** objects

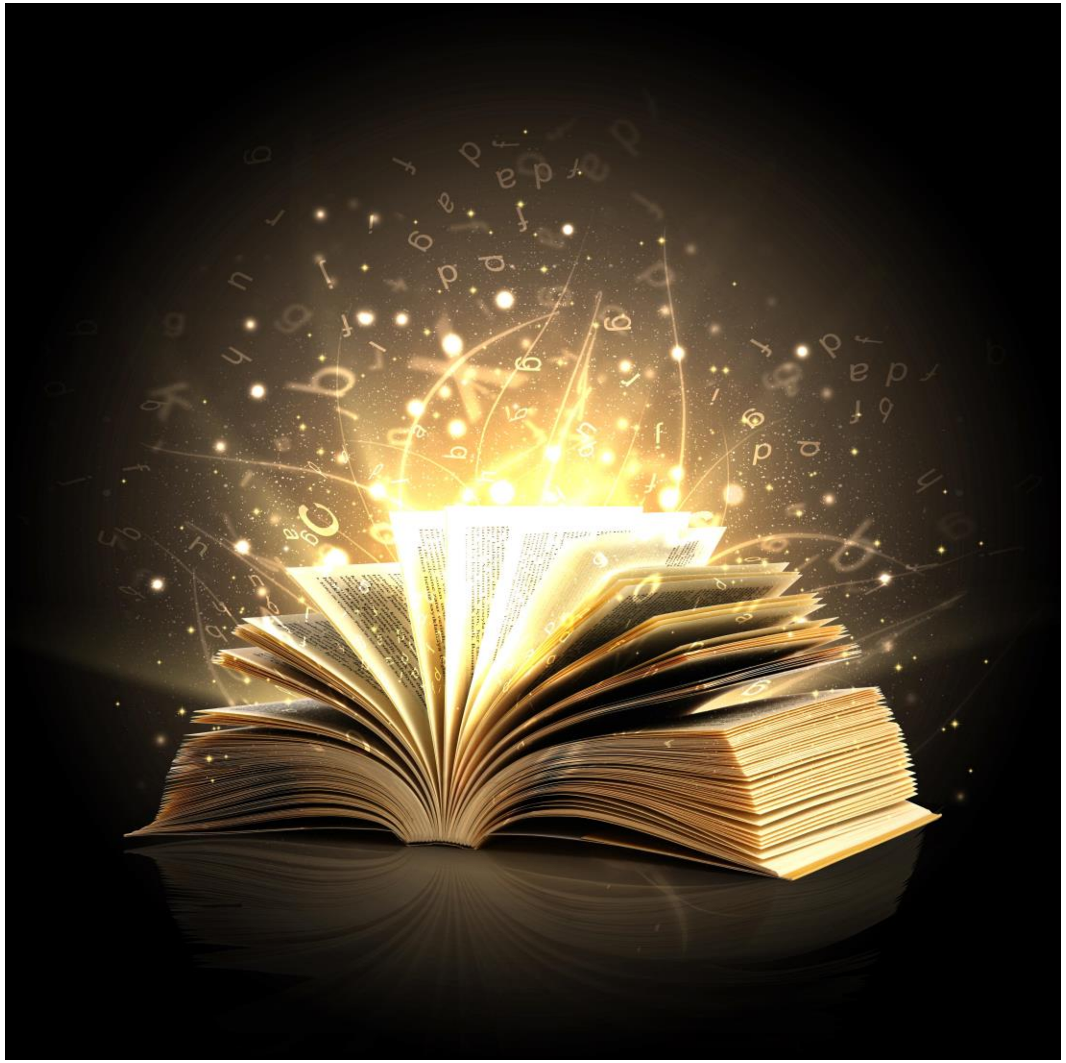
- Normal objects behave according to the rules of common sense and follow the laws of traditional physics



In the world there are **normal** and **quantum** objects

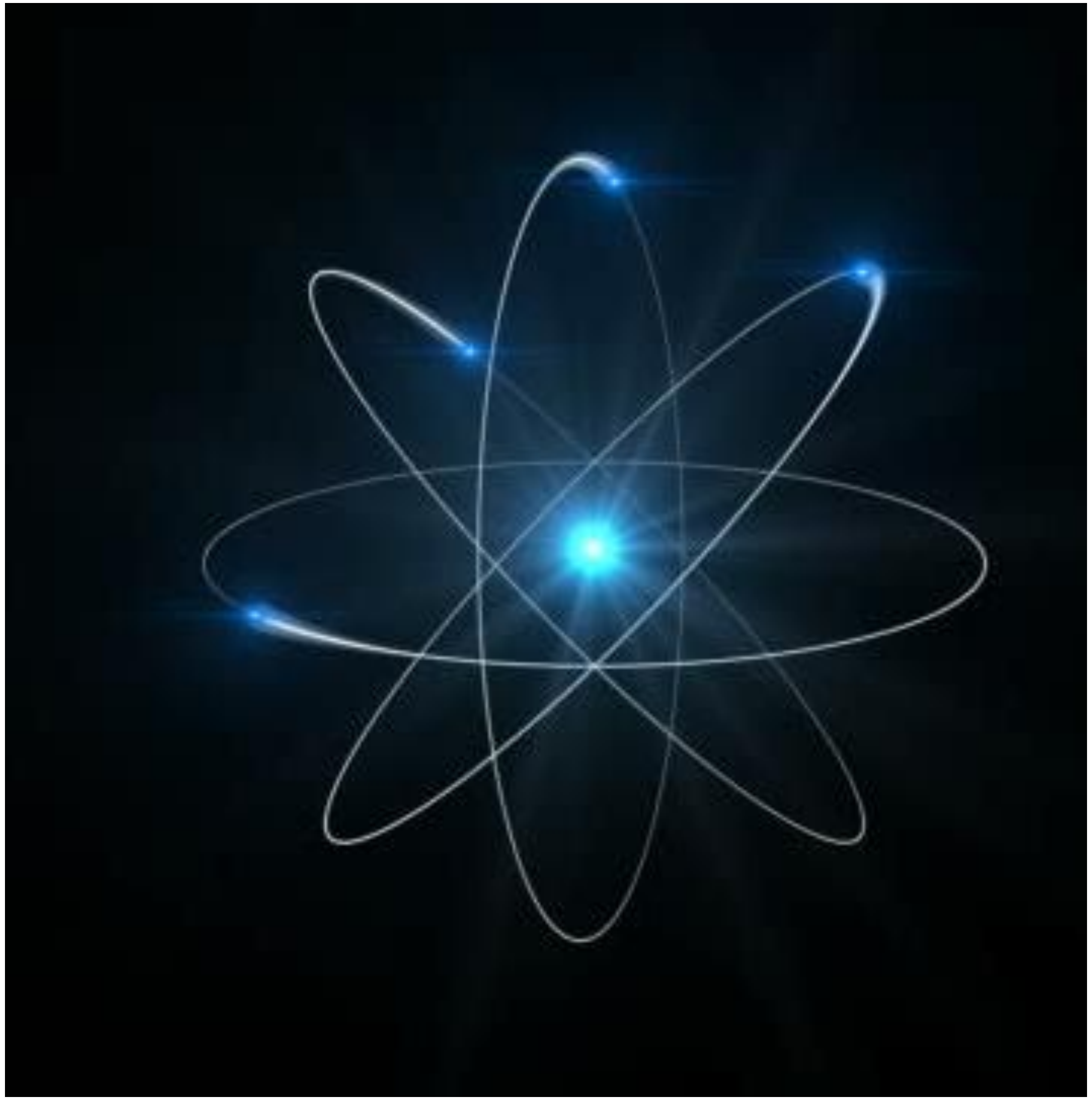
- Normal objects behave according to the rules of common sense and follow the laws of traditional physics

- Quantum objects behave in a funny and strange way, because they follow the laws of quantum physics



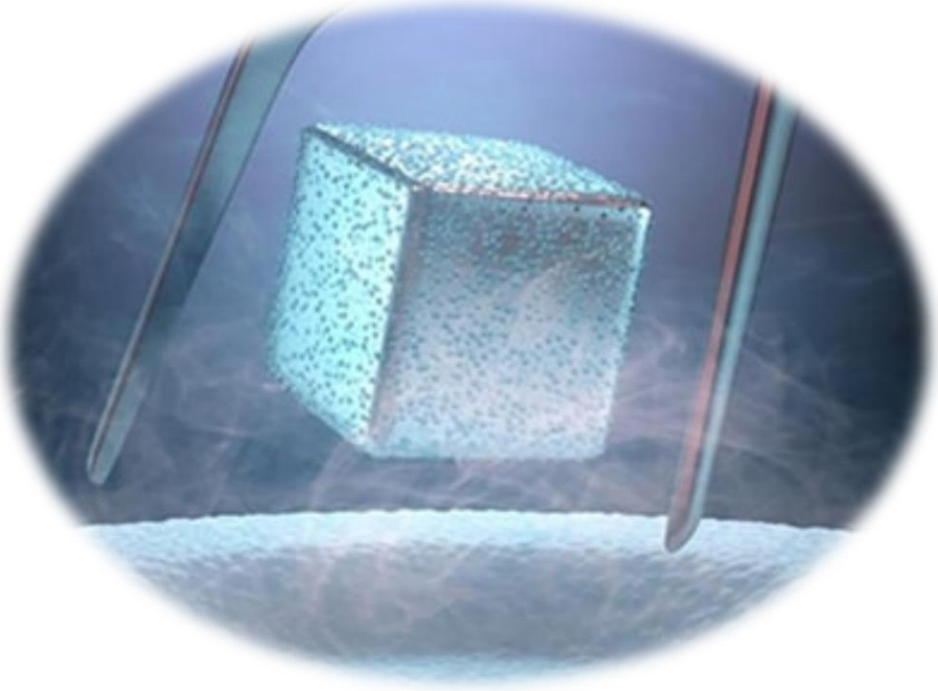
But what are quantum objects?

- Very small, microscopic particles, such as atoms, electrons, photons, if isolated from the rest of the world, behave like quantum objects



But what are quantum objects?

- Very small, microscopic particles, such as atoms, electrons, photons, if isolated from the rest of the world, behave like quantum objects
- Superconducting objects, cooled to temperatures very close to absolute zero, behave like quantum objects



What is a Quantum Computer?

A computer composed of quantum objects called

qubits

which follows the laws of

quantum mechanics

We perform computations by **manipulating** qubits

How is a qubit made?

Technology	*Operation*
Superconductors	20 mK
Photons	1 K
Electrons	1 K
Ions	High vacuum
Atoms	High vacuum
Diamonds	Environment
Topological	…

Principles of Quantum Mechanics

Principles of Quantum Mechanics

Principles of Quantum Mechanics

Superposition

Principles of Quantum Mechanics

Principles of Quantum Mechanics

Superposition

Entanglement

Principles of Quantum Mechanics

Principles of Quantum Mechanics

Superposition

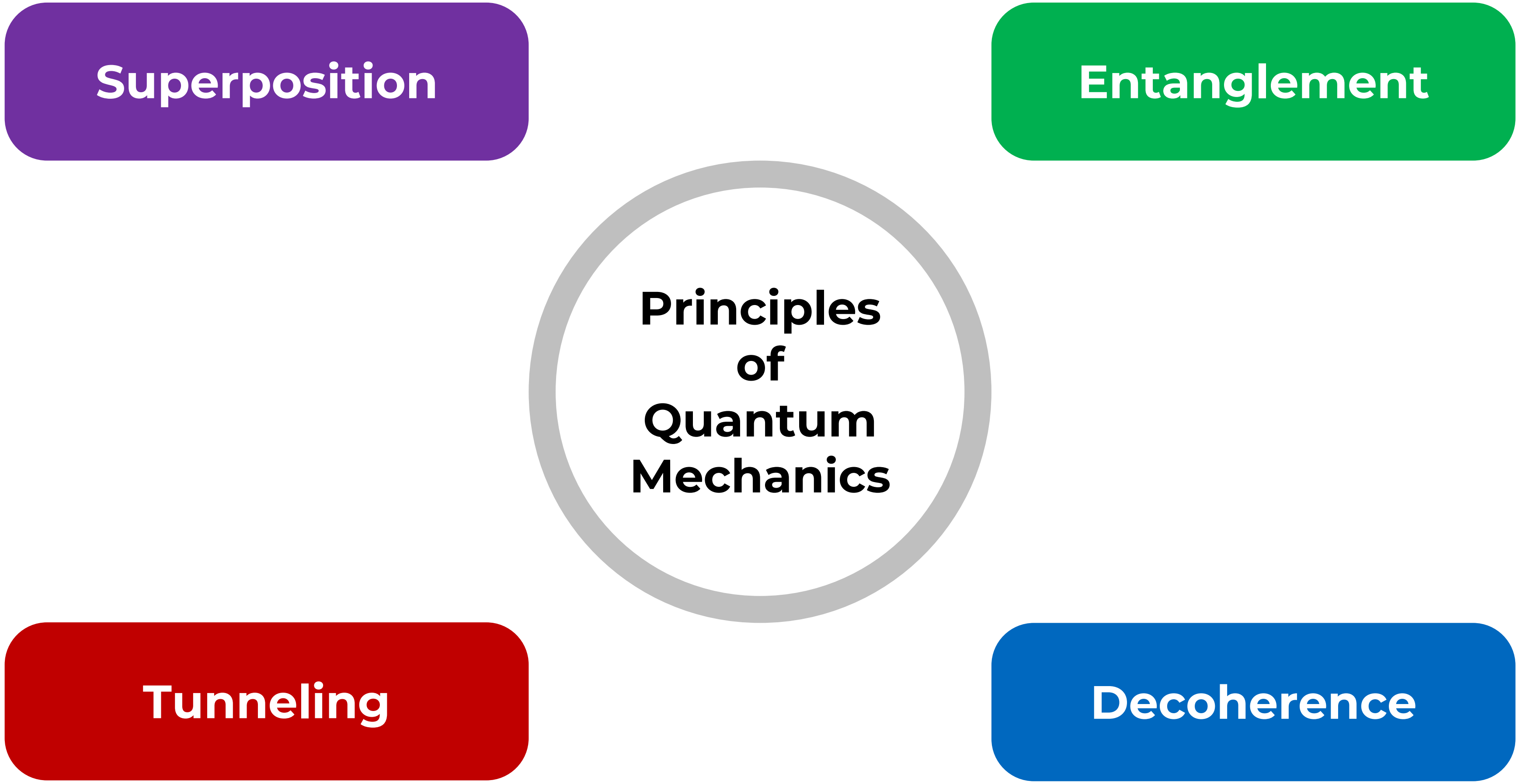
Entanglement

Principles of Quantum Mechanics

Tunneling

13:20

Principles of Quantum Mechanics



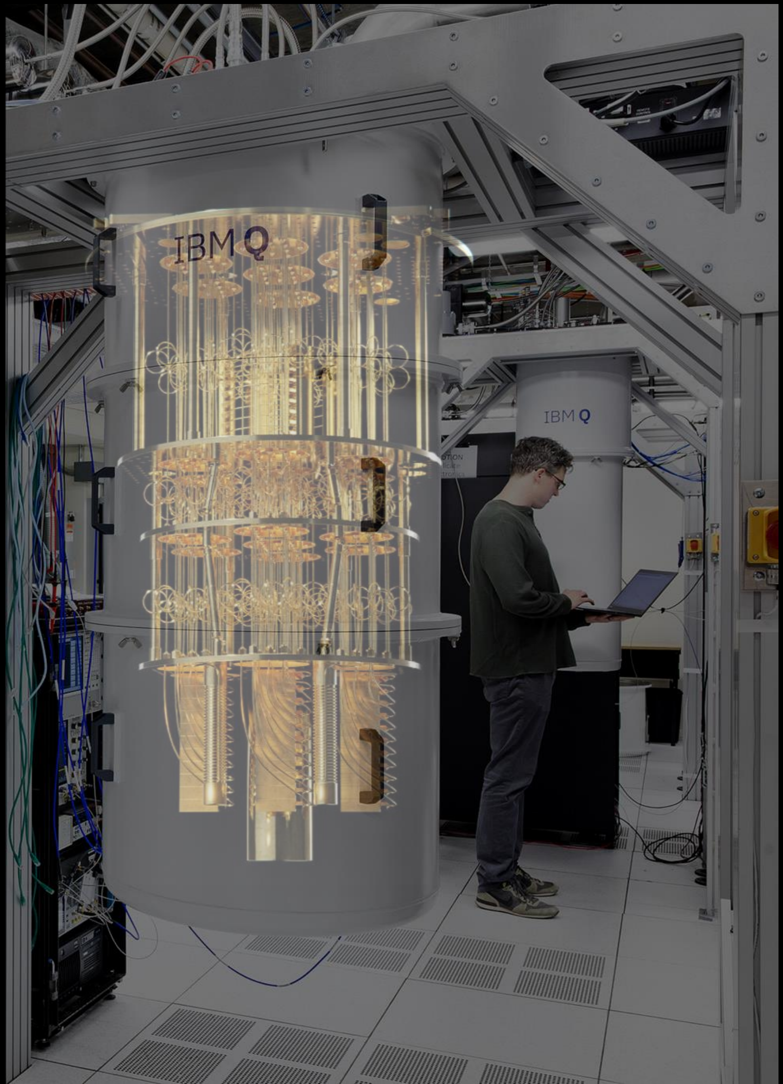
What makes a Quantum Computer fast?

Classical Computing	Quantum Computing
With **n bits** you can run up to **n operations** at the same time	With **n qubits** you can run up to **2^n operations** at the same time

How is a Quantum Computer made?

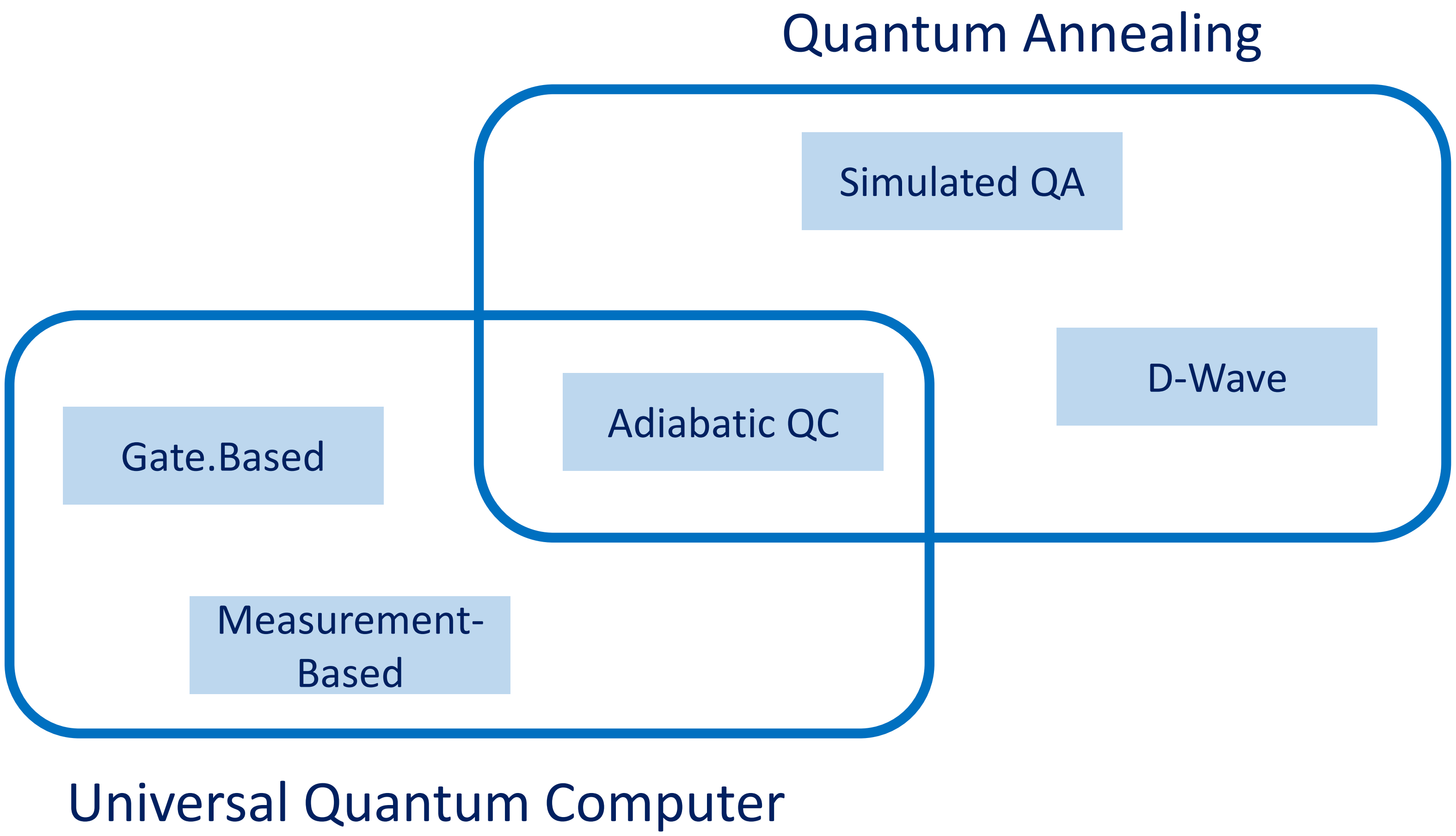
13:20

How is a Quantum Computer made?

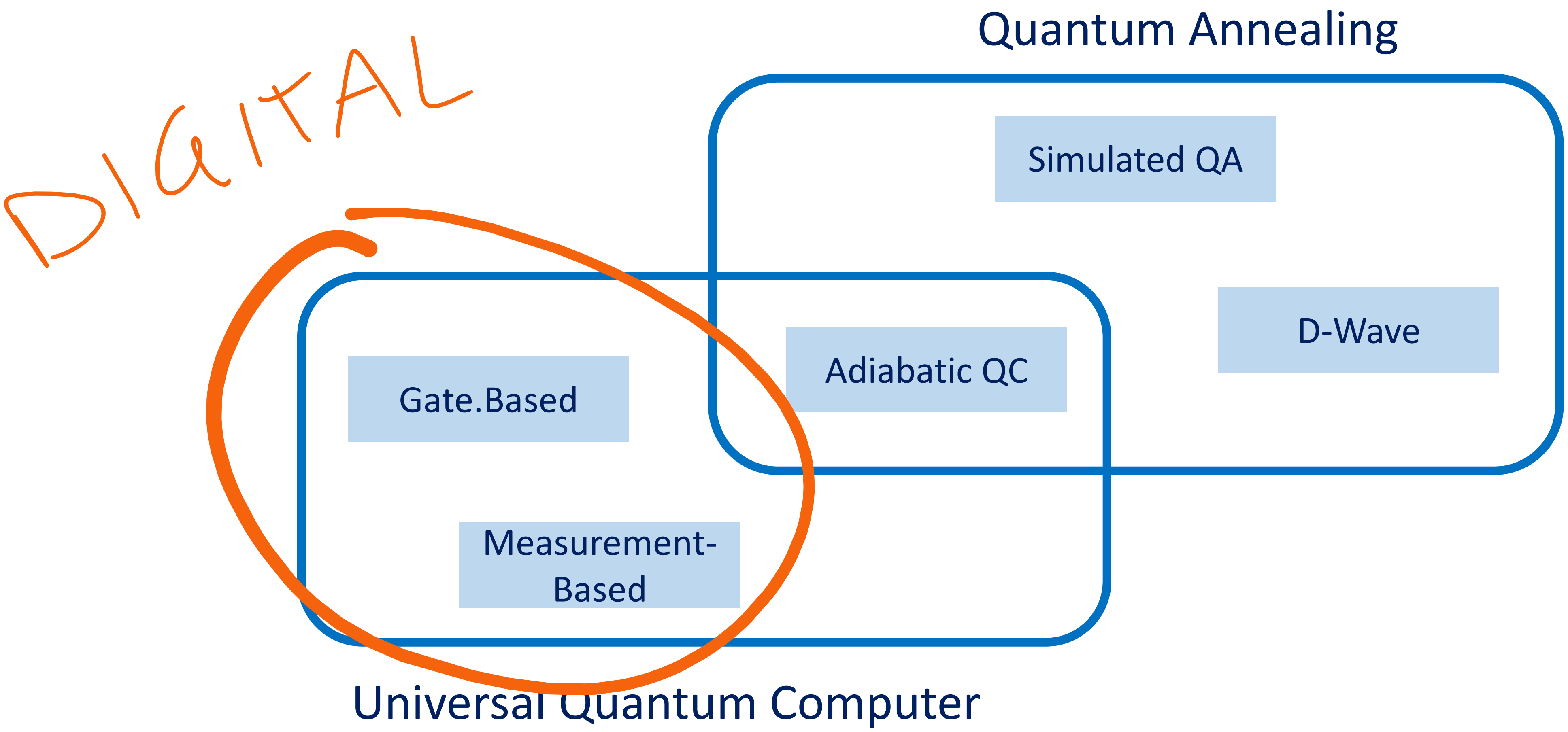


How is a Quantum Computer made?

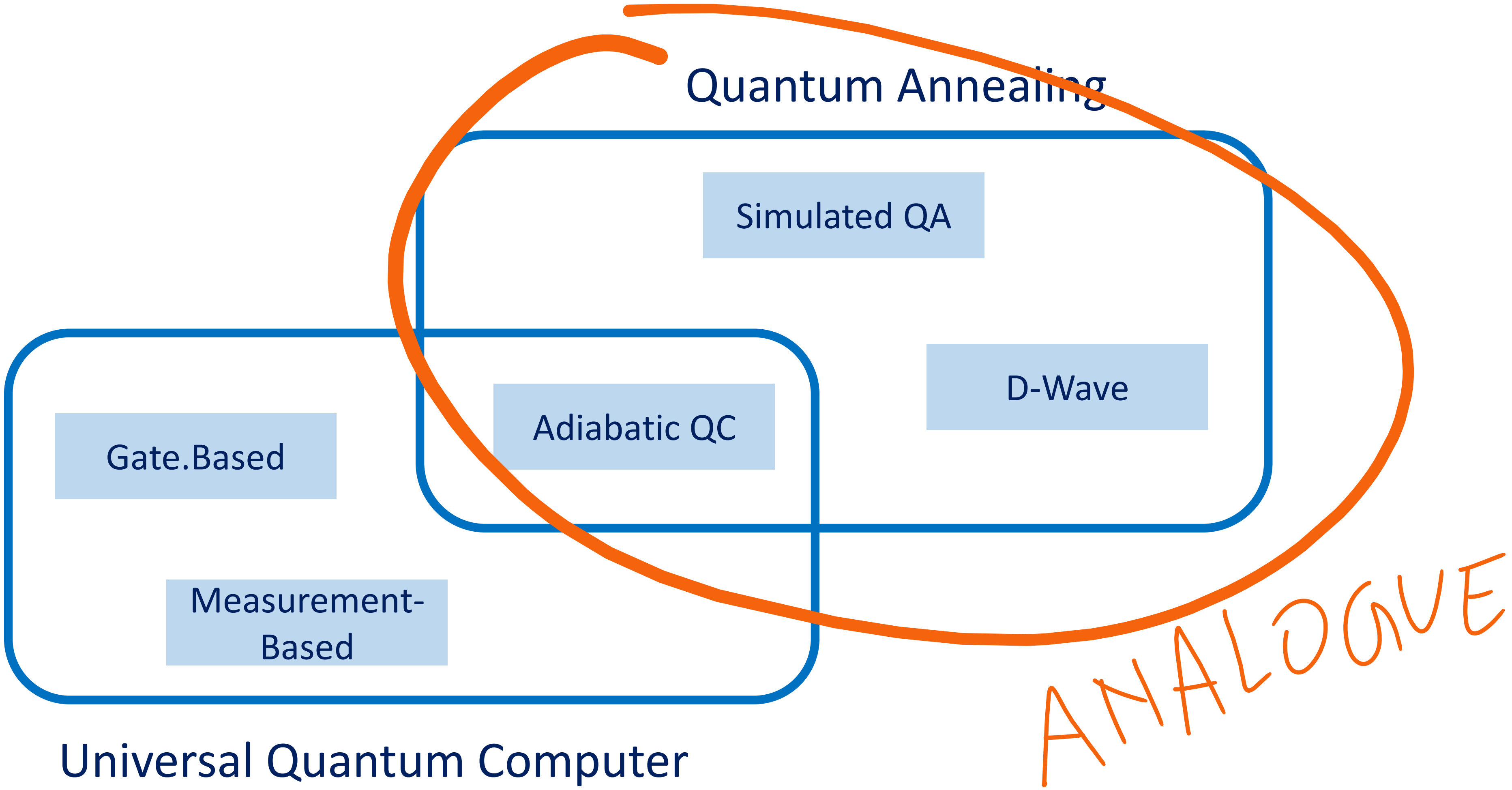
Quantum Computing Models and Architectures: how do you manipulate qubits …



Quantum Computing Models and Architectures: how do you manipulate qubits …



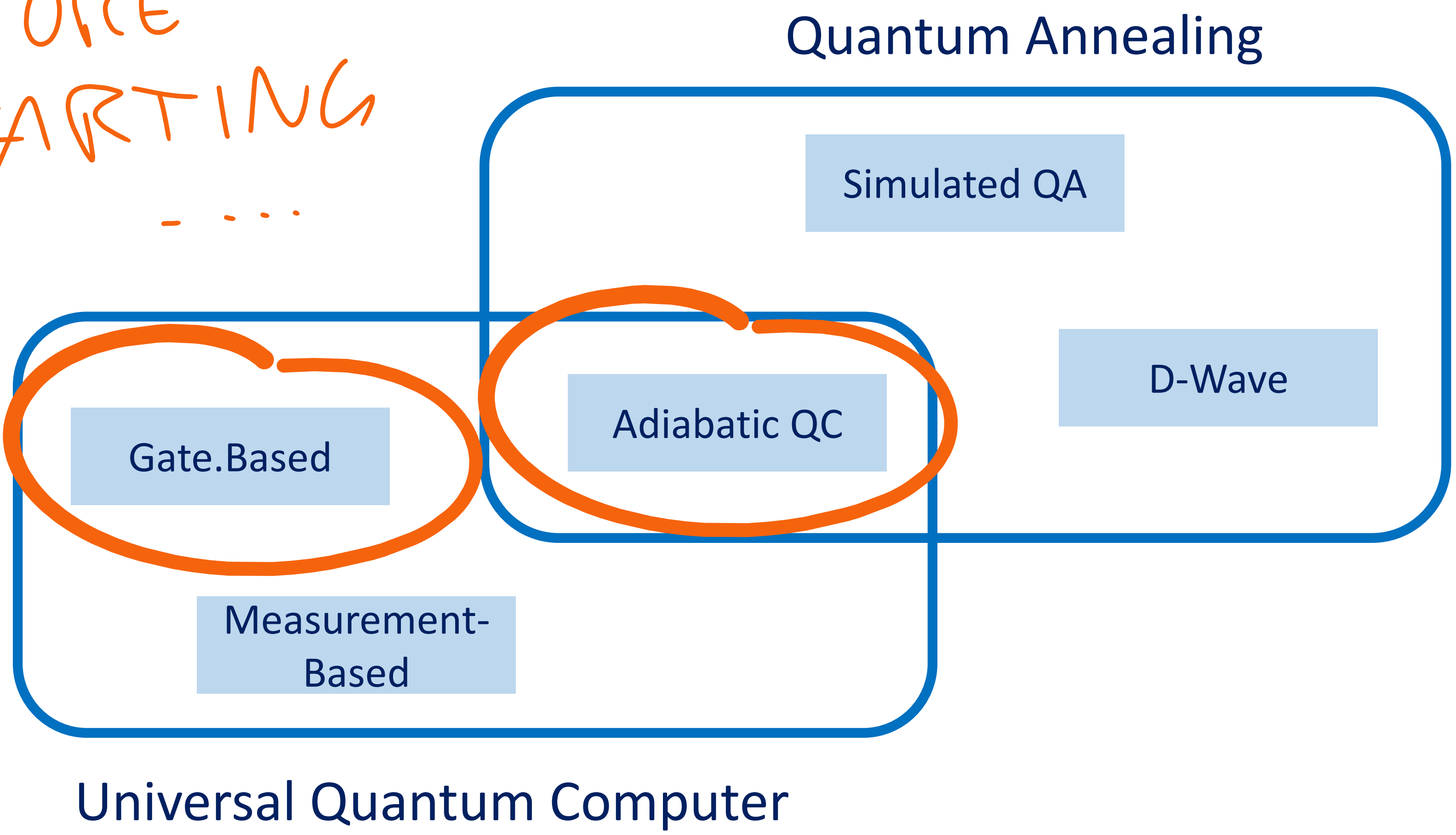
Quantum Computing Models and Architectures: how do you manipulate qubits …



13:20

Quantum Computing Models and Architectures: how do you manipulate qubits …

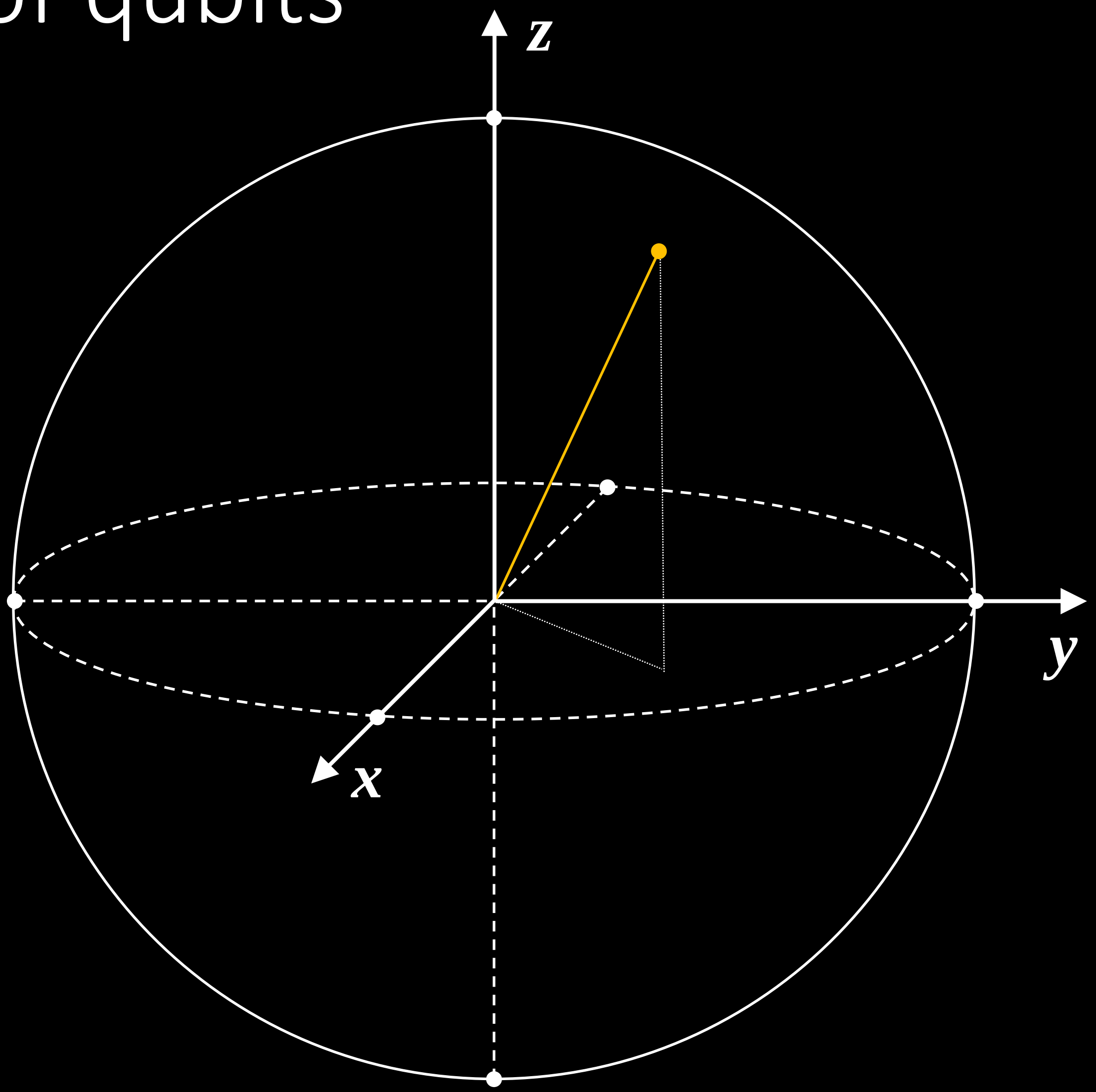
BEFORE STARTING
- . . .



(Scary) Introduction to Quantum Computing

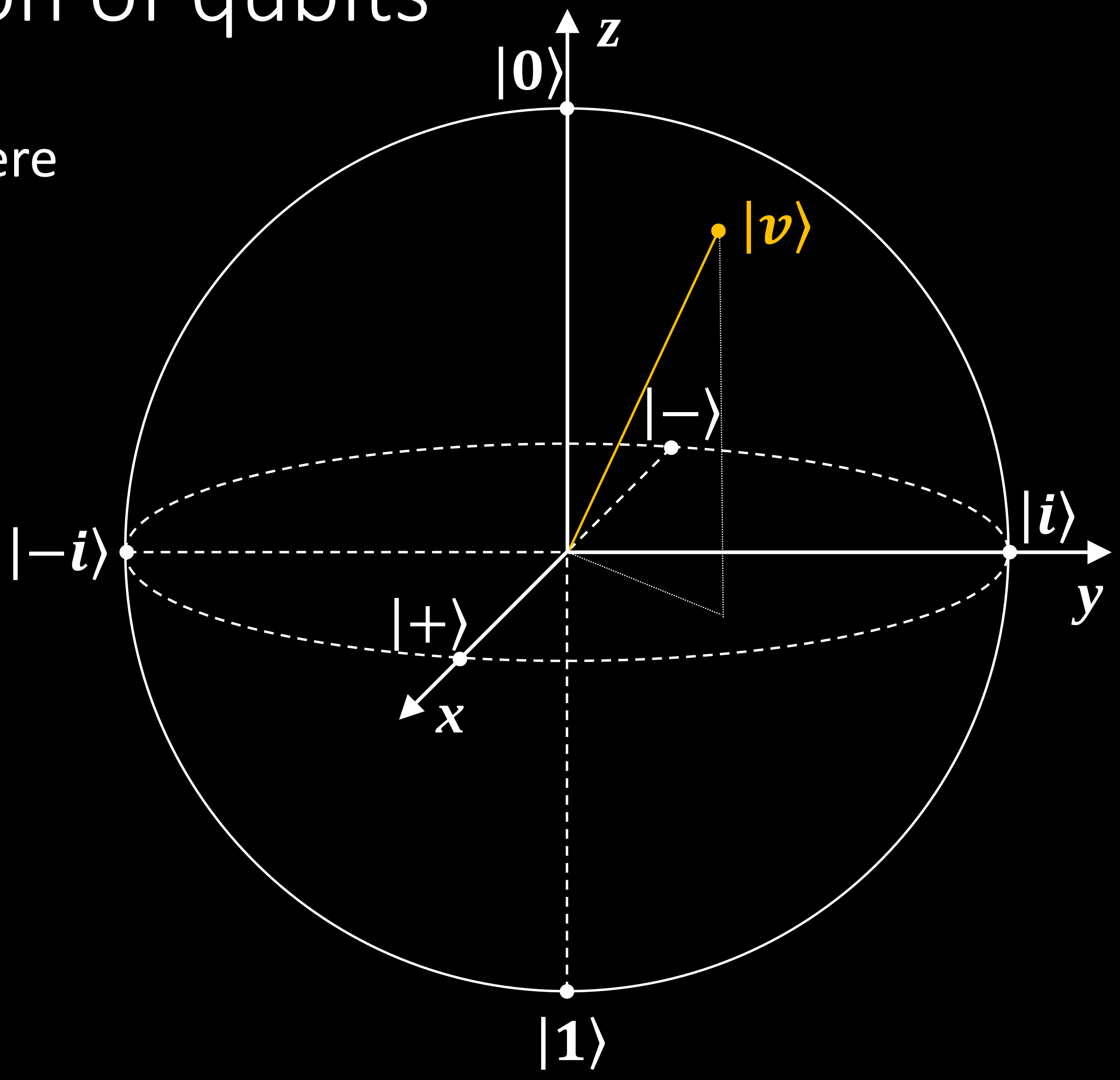
Bloch sphere representation of qubits

- Qubits can take infinite values on the sphere



Bloch sphere representation of qubits

- Qubits can take infinite values on the sphere
- Special values for qubits
- $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
- $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
- $|+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$
- $|-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$



Dirac’s notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos

Dirac's notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Dirac's notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Dirac’s notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Dirac’s notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Dirac's notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Dirac’s notation:

- Writing QC algorithms mostly reduce to manipulate **unitary** matrices and vectors (of complex numbers)
- Dirac's notation (also know as **bra-ket** notation): compact notation for the most common manipulations that happens in QC algos
- **ket** = $|v\rangle \leftrightarrow \vec{v}$ (column vector)
- **bra** = $\langle v| \leftrightarrow \vec{v}^H$ (row vector)
- $\langle x|y\rangle \leftrightarrow \vec{x}^H \cdot \vec{y}$ (scalar product between vectors $\vec{x}$ and $\vec{y}$)
- $a|v\rangle \leftrightarrow a \cdot \vec{v}$ (product between constant a and column vector $\vec{v}$)
- $M|v\rangle \leftrightarrow M \cdot \vec{v}$ (product between square matrix M and column vector $\vec{v}$)

- $\langle x|M|y\rangle \leftrightarrow \vec{x}^H \cdot M \cdot \vec{y}$

Adiabatic Quantum Computing & Quantum Annealing

Adiabatic Quantum Computing (AQC)

- Instead of building a circuit to do what we want, one operation at a time, we leverage the natural tendency of a physical system to evolve towards (and remain into) a state of minimal energy
- Adiabatic process
  - a process occurring without transferring energy or mass with the systems surroundings
- The adiabatic theorem for the evolution of a quantum system states that
  - if there is an **energy gap** between the ground state and other states

    and
  - if the evolution of the system in time is **sufficiently slow**

    then
  - the system remains in a state of minimal energy (ground state)

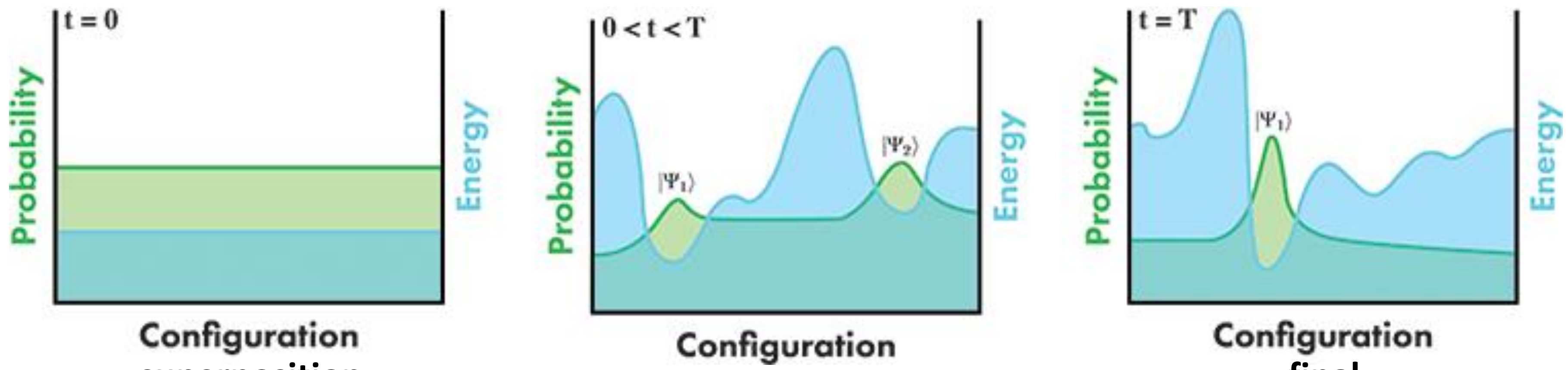
AQC: The idea

- I have a problem to solve which I can represent as the energy of a quantum system called the Hamiltonian
  - the minimum energy state (**ground state**) corresponds to the **solution** I want
  - it is difficult to find
- I can evolve a quantum system maintaining it in a minimum energy state

adiabatic condition: slow evolution

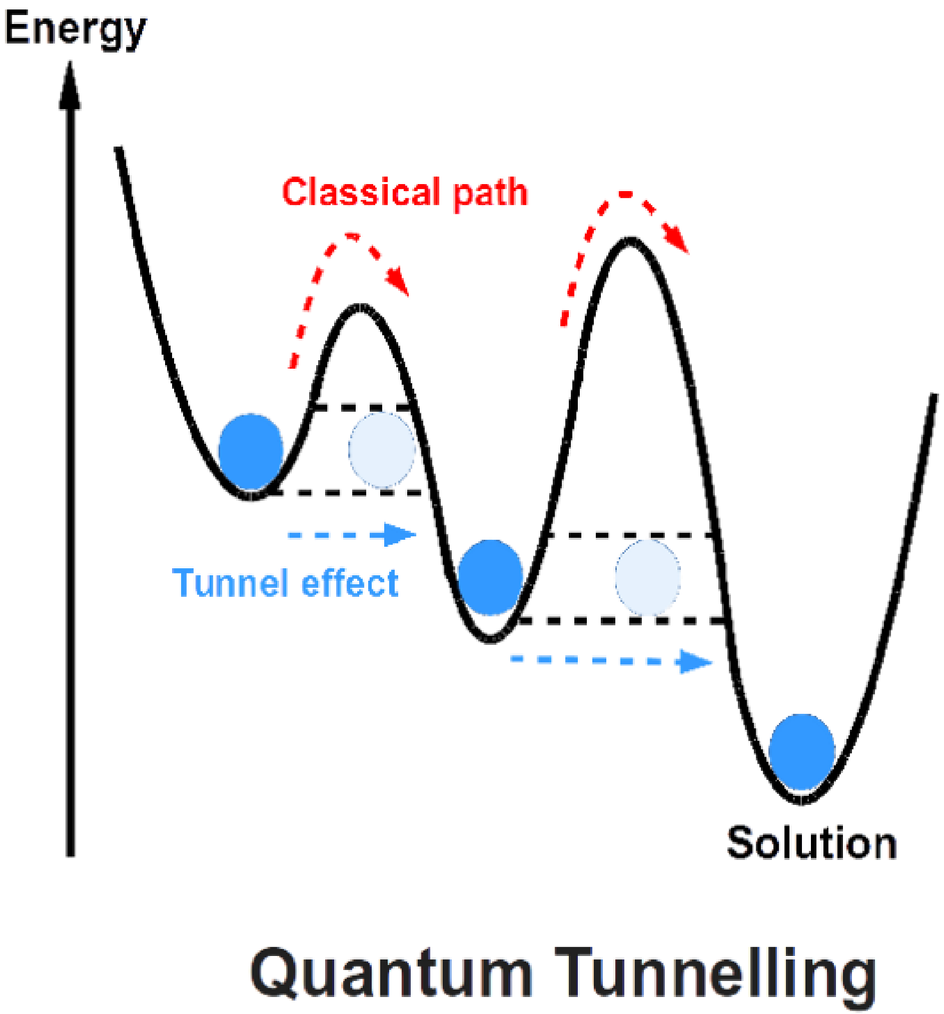
AQC: Procedure

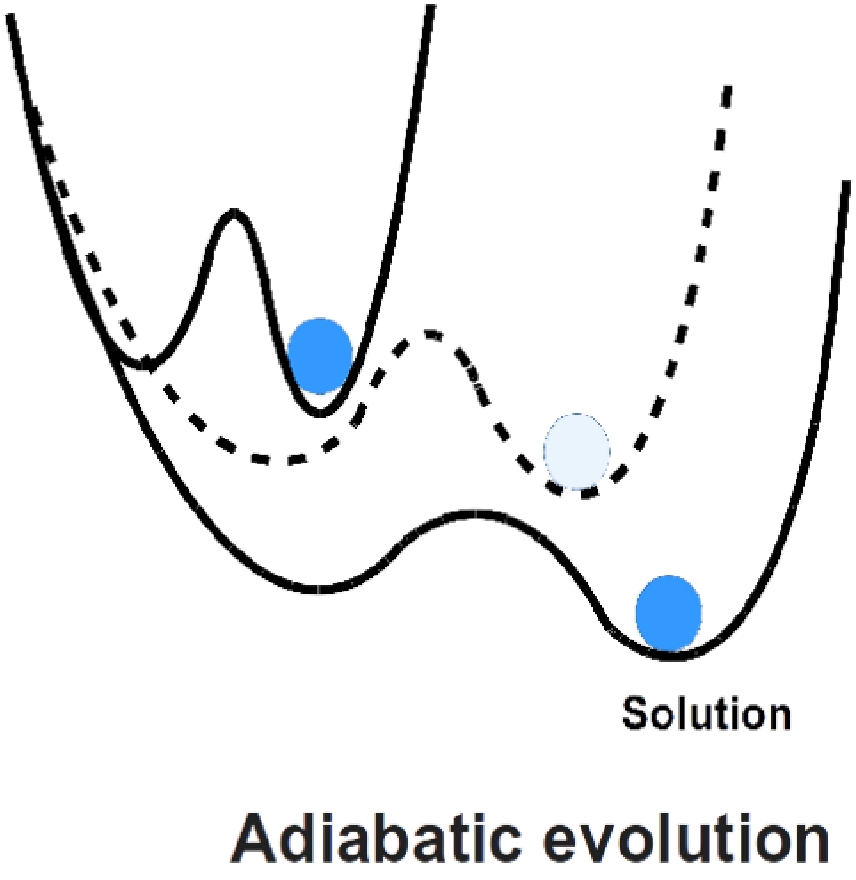
- Start from an **initial** configuration (**superposition**) in which we can easily find the ground state
- **Slowly** alter the system to include the problem I want to solve while reducing the weight of the initial configuration
- Once the initial configuration weight goes to zero, the system only depends on the problem I want to solve, and it remains in the ground state



superposition **final**

Quantum Annealing vs. Adiabatic Evolution



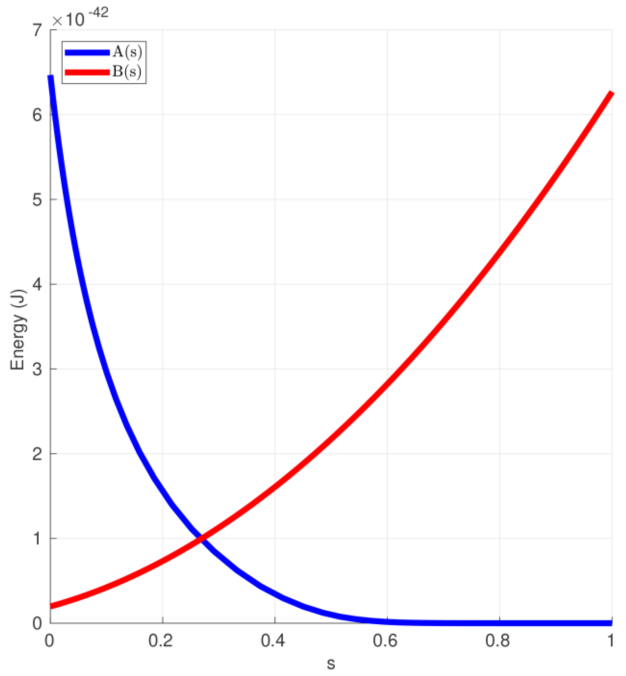


Time-dependent Hamiltonian

- The energy of a quantum system can be represented by a time dependent Hamiltonian
- which is composed by two terms

$$H(t) = A(t)H_A + B(t)H_B$$

- Here H_A and H_A represent the two Hamiltonians for the initial state (A) and the problem that we want to solve (B)
- $\boldsymbol{A(t)}$ and $\boldsymbol{B(t)}$ control the weight of the two Hamiltonians over time
  - analogous to the temperature schedule used in Simulated Annealing



Ising model and QUBO problems

- Currently available QA hardware uses Hamiltonian that describes an Ising model
- The Ising model describe the interactions between qubits z

$$H_B = \sum_i h_i z_i + \sum_{i>j} J_{i,j}\, z_i z_j$$

- where h_i represents a bias on qubit *i* and $J_{i,j}$ describes the coupling strength between qubits *i* and *j*
- A problem can be described by setting the values of h_i and $J_{i,j}$
- The Ising Hamiltonian describes a quadratic unconstrained binary optimization problem (**QUBO**)

Ising model and QUBO problems

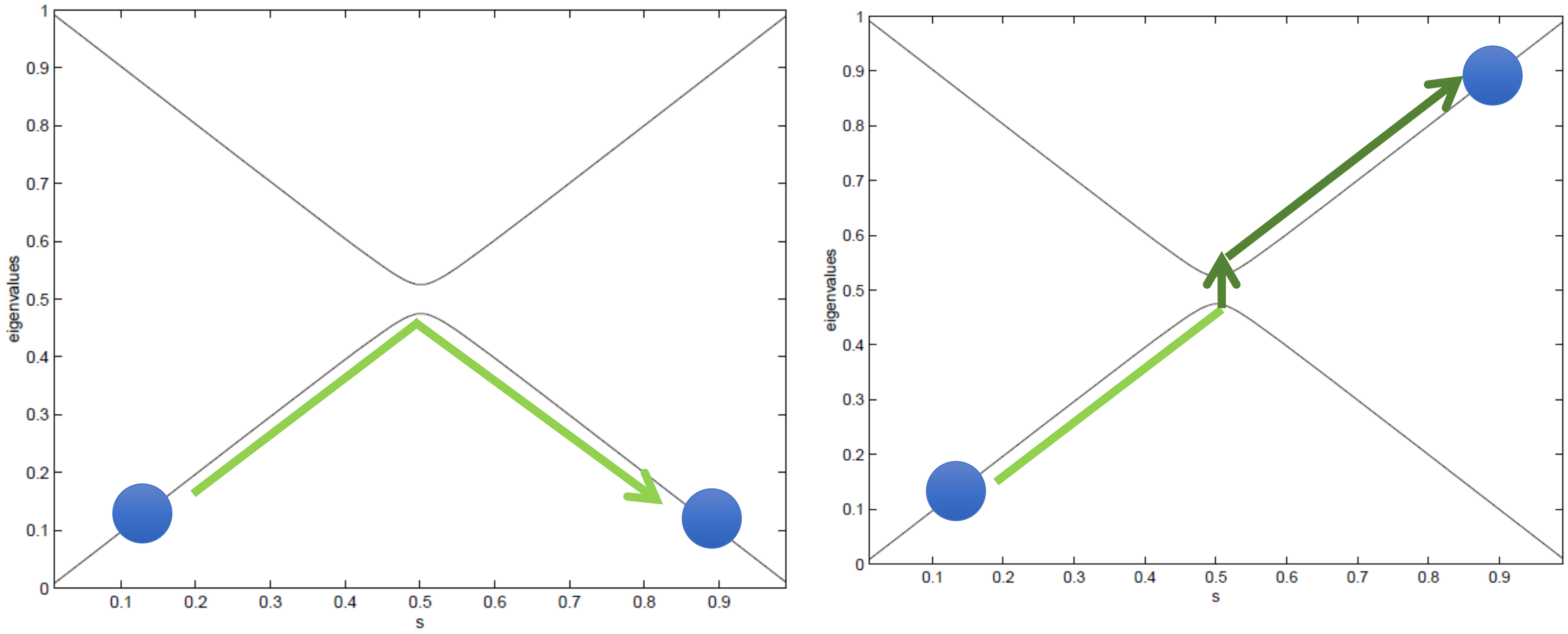
- With a change of variables, it is possible to show that the Ising problem is equivalent to a Quadratic Unconstrained Binary Optimization problem (**QUBO**)

$$\operatorname*{argmin}_{\boldsymbol{x}} \left(\sum_{i} a_i \, x_i + \sum_{i \geq j} Q_{i,j} \, x_i x_j \right) = \operatorname*{argmin}_{\boldsymbol{x}} \boldsymbol{x}^T Q \boldsymbol{x}$$

- The problem is often represented in terms of spins $s_i \in \{-1, +1\}$
- We can easily transform the problem formulation from spins $s_i \in \{-1, +1\}$ to binary variables $x_i \in \{0,1\}$

$$x_i = \frac{1}{2} s_i + \frac{1}{2}$$

The Eigenspectrum



Remains in the ground state

Jumps into a higher energy state
"Landau–Zener" transition
due to heat or evolution too fast

AQC vs. Quantum Annealing

- Adiabatic QC leverages quantum tunneling, just as Quantum Annealing, and in general has a wider computational power

- Adiabatic convergence is stricter than QA, so the first implies the latter
  - adiabatic QC is a specific type of QA, which is also universal

- Quantum Annealing evolves the same Hamiltonian but relaxes some of the stringent requirements of the adiabatic theorem.

Quantum Annealing and D-Wave

- Adiabatic Quantum Computing is equivalent to quantum circuit model (universal)
- D-Wave QPU
  - **non-positive real off-diagonal elements** of the Ising formulation *J*
  - as such, is not not universal
- D-Wave QPU
  - coupling every qubit to every other qubit is physically impractical
  - *J* must be very sparse …
- One of the most immediate consequences is that we cannot rely on a single measurement, but we need to run the experiment multiple times to account for the impact of both the noise and the limited evolution schedule
- We use QA to do **sampling** from a “distribution”

Thanks