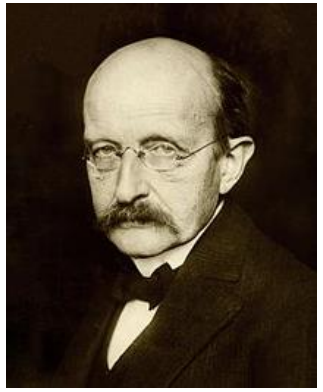
QUANTUM COMPUTING
TECHNOLOGY OF THE FUTURE?

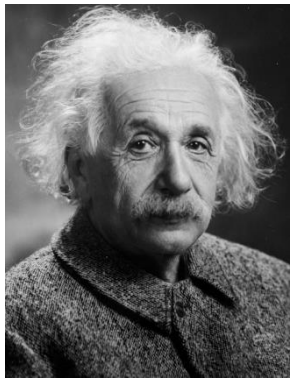
23. FEBRUAR 2022 I DR. DENNIS WILLSCH

WOHER KOMMT DER QUANTENCOMPUTER?

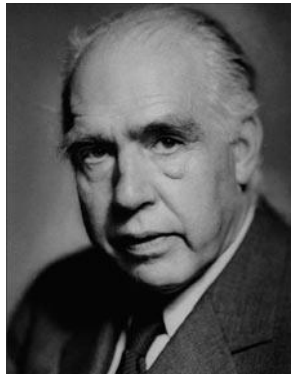
Ab 1900: Entstehung der Quantentheorie



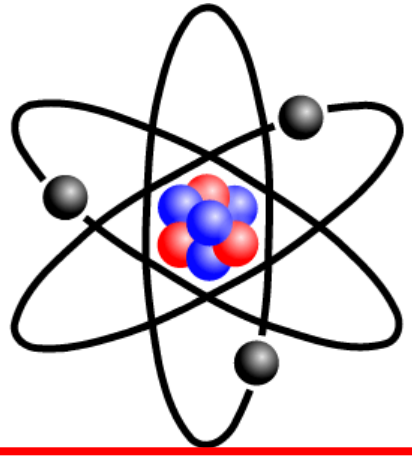
Planck



Einstein



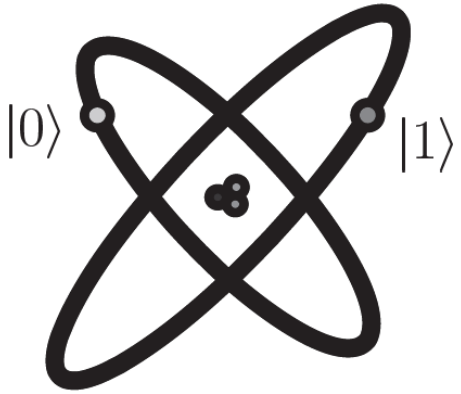
Bohr



1959, 1982, 1994: Idee: Quanteneffekte für Computer nutzen

Feynman

Shor

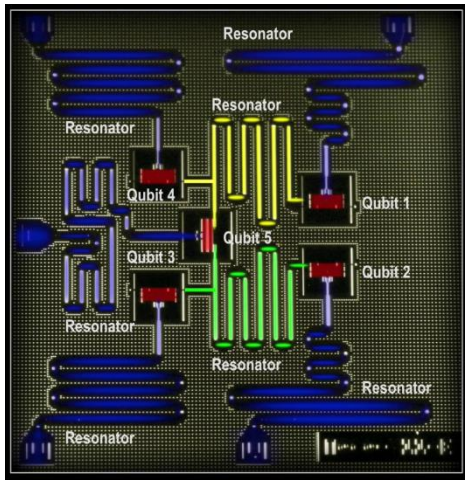


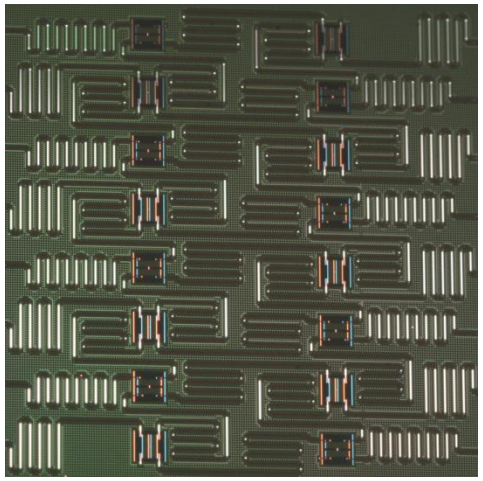
2007 - heute: Erste Quantencomputer(-Prototypen) verfügbar

QUANTENCOMPUTER

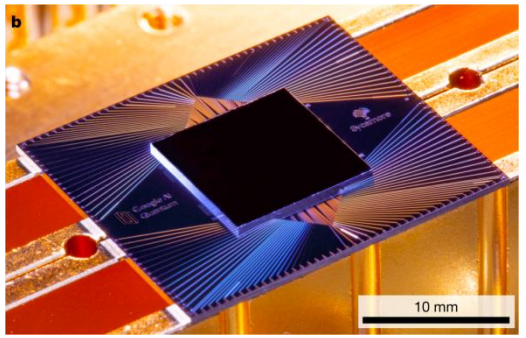
Es gibt 2 Typen QC: Gatterbasierte QC und Quantenannealer

IBM: 1, 5, 7, 16, 27, 65, 127 Qubits

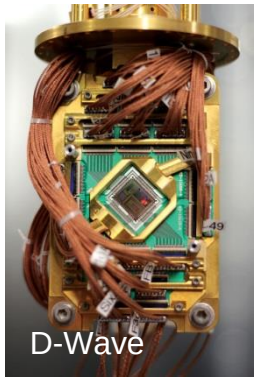


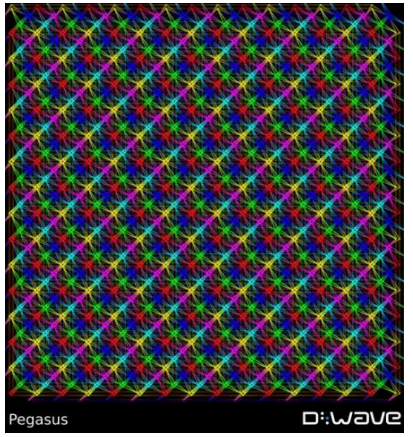


Google: 53, 72 Qubits

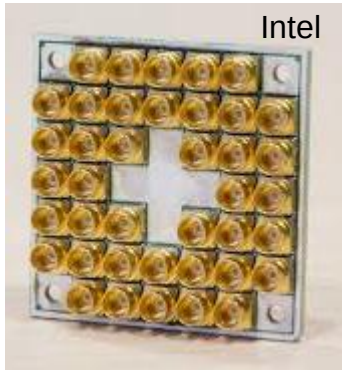


D-Wave: 2048, 5760 Qubits

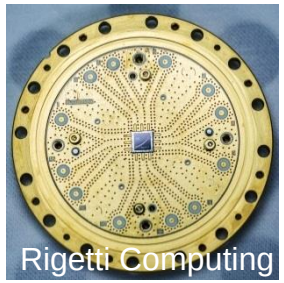




Intel: 17, 49 Qubits



Rigetti Computing: 8, 16, 19 Qubits

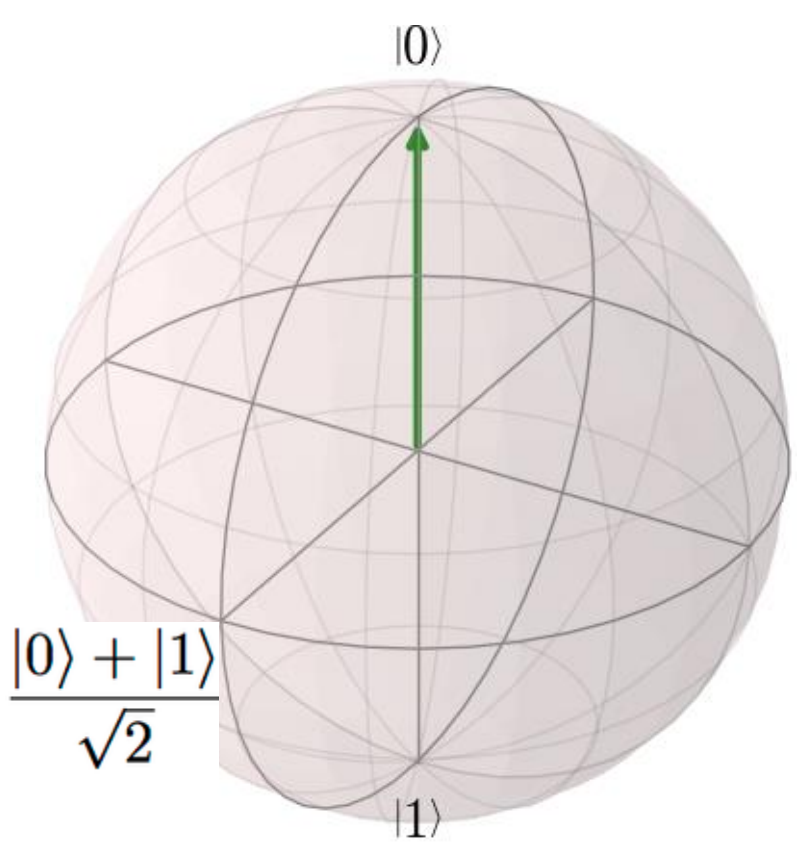


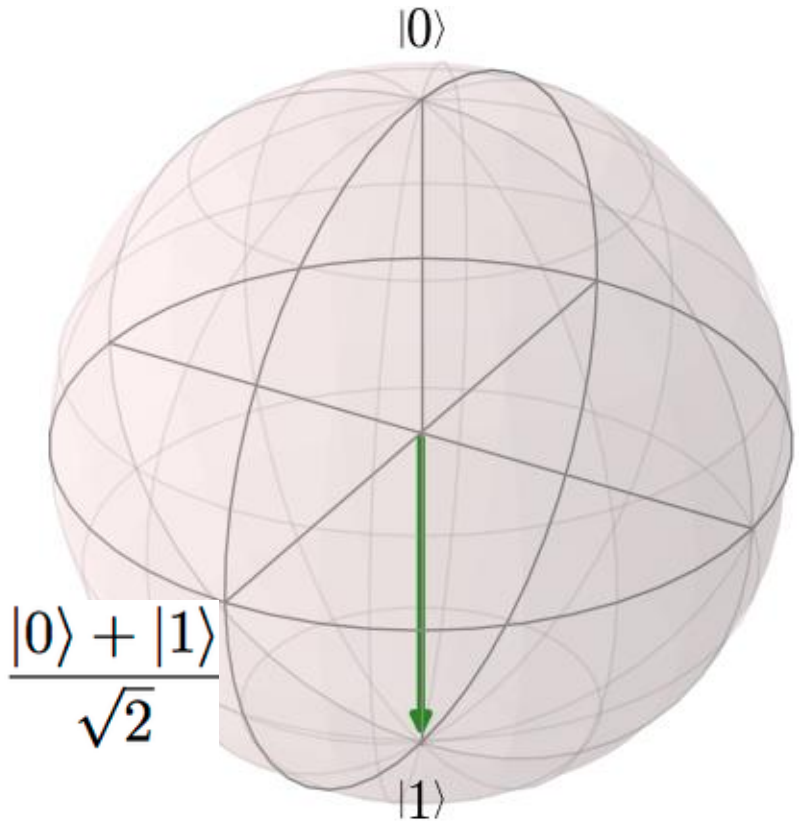
WAS IST EIN QUBIT?

Digitaler Computer: Bit

Quantencomputer: Quantum Bit = Qubit

$$\alpha|0\rangle + \beta|1\rangle$$

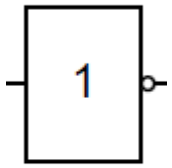




WELCHE OPERATIONEN HAT EIN GATTERBASIERTER QC?

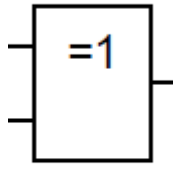
Digitale Gatter:

1 bit: NOT



a	y
0	1
1	0

2 bits: XOR



a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

und viele weitere

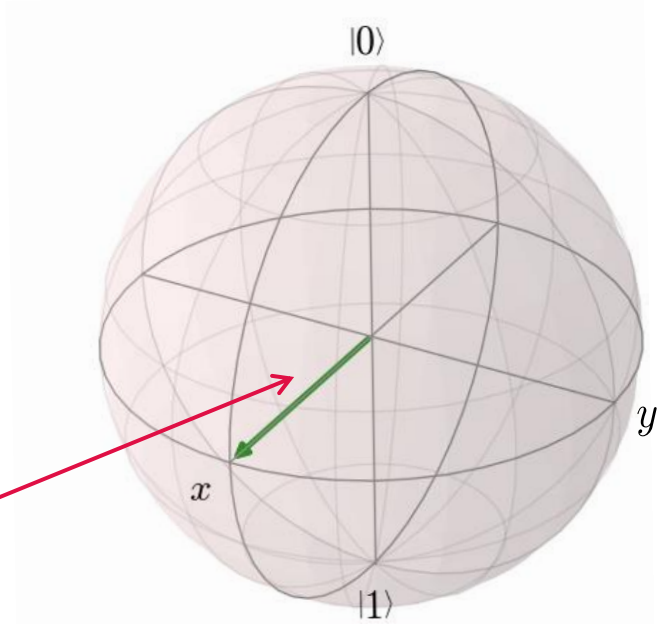
Quantengatter:

1 qubit: $\alpha|0\rangle + \beta|1\rangle$

X-Gatter (NOT): $|0\rangle$ — X — $|1\rangle$

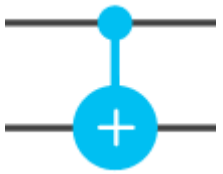
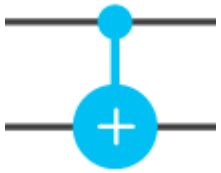
Y-Gatter (i-NOT): $|0\rangle$ — Y — $i|1\rangle$

Hadamard: $|0\rangle$ — H — $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$



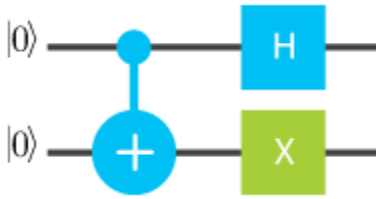
und viele weitere

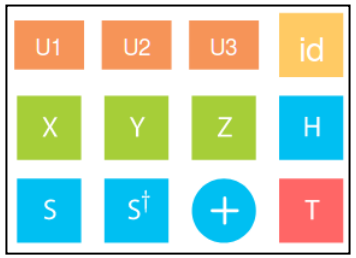
2 qubits: $\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$

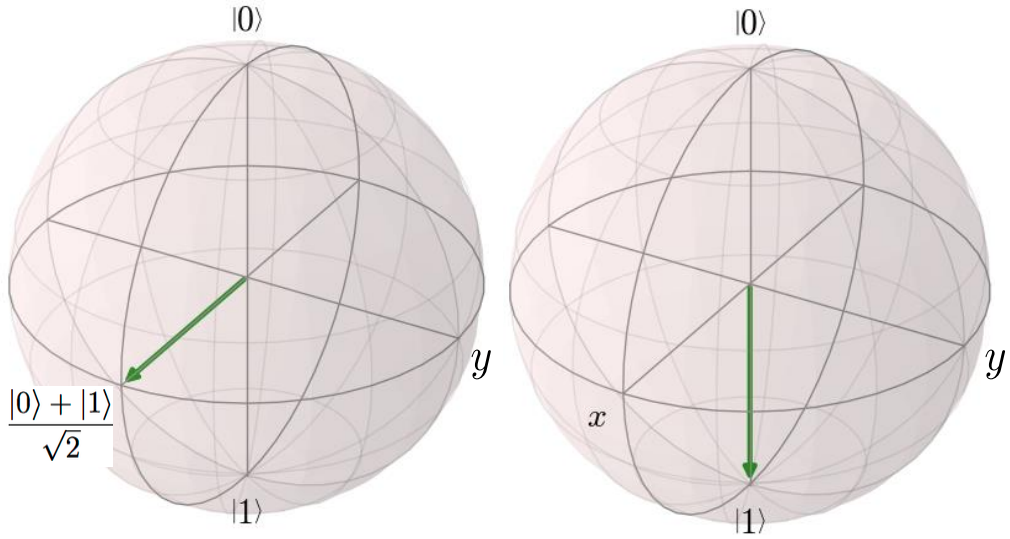
CNOT: $|00\rangle$  $|00\rangle$ $\quad$ $|10\rangle$  $|11\rangle$

WIE BENUTZT MAN EINEN GATTERBASIERTEN QC?

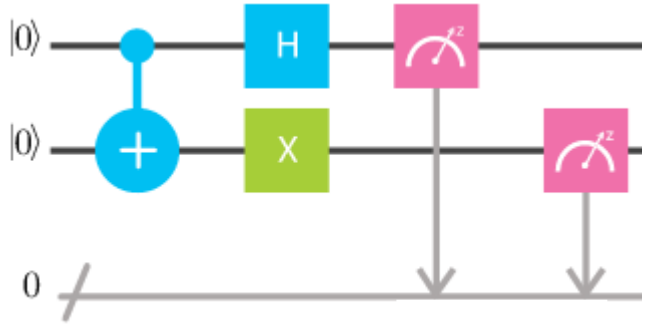
- **Baue Schaltungen aus Quantengattern**







- **Messe Qubits → Ergebnis: Bits!**



→ messe zu 50% **0** und zu 50% **1**

→ messe zu 100% **1**

→ Ausgabe 50/50: **„01“ oder „11“**

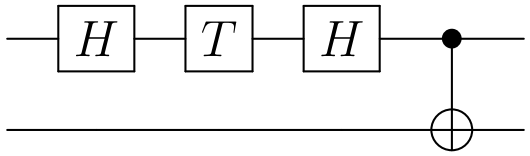
$$\sqrt{0.50}|01\rangle + \sqrt{0.50}|11\rangle$$

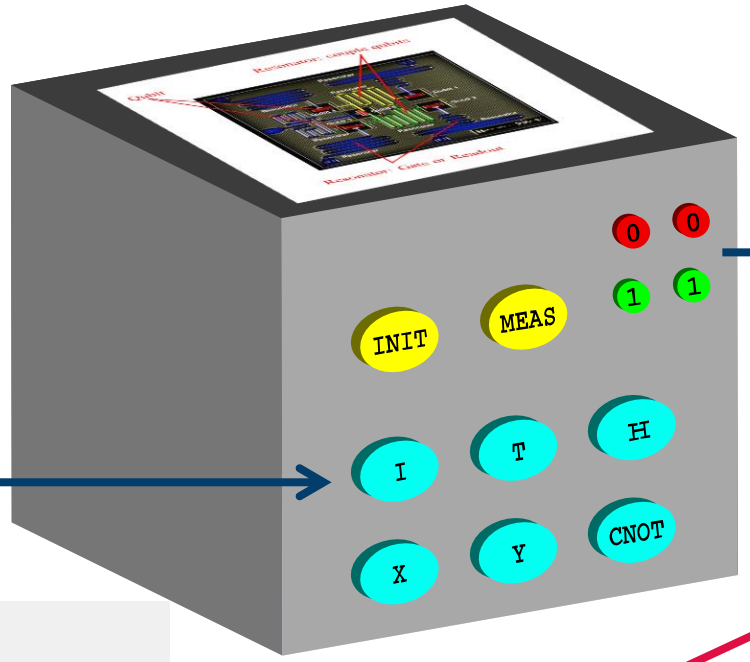
- **Wiederhole Programm so oft wie nötig**

GATTERBASIERTE QC: PROGRAMMIERUNG

Lecture Notes:
Programming Quantum Computers
https://arxiv.org/pdf/2201.02051.pdf

Eingabe:
Programm =
Schaltung aus
Quantengattern





Ausgabe: Bitstrings
„00“, „01“, „10“, „11“

```
1  import qiskit
2
3  circuit = qiskit.QuantumCircuit(2)
4  circuit.h(0)
5  circuit.t(0)
6  circuit.h(0)
7  circuit.cx(0,1)
8
9  from juqcs import Juqcs
10 backend = Juqcs.get_backend('statevs
```

JUQCS
(Jülich Universal Quantum Computer Simulator)

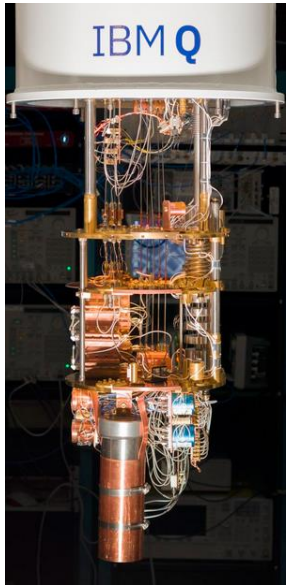
$$\sqrt{0.85\ldots}|00\rangle + \sqrt{0.15\ldots}|11\rangle$$

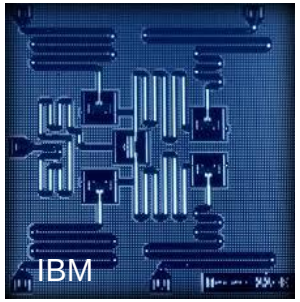
messe zu 85 % **00**

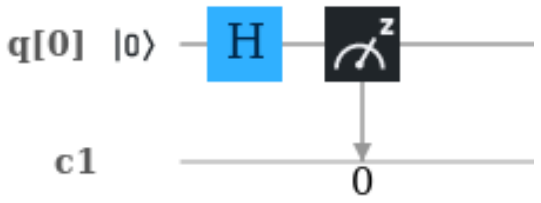
und zu 15 % **11**

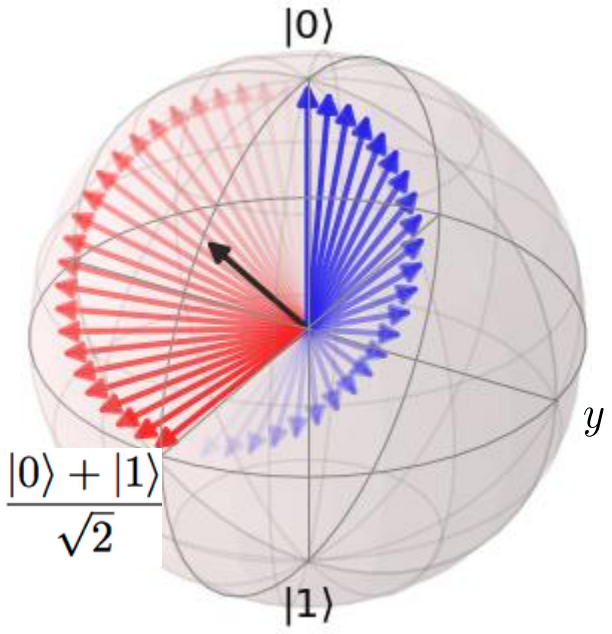
GATTERBASIERTE QC: ANWENDUNGEN

Münzwurf: 1 faire Münze

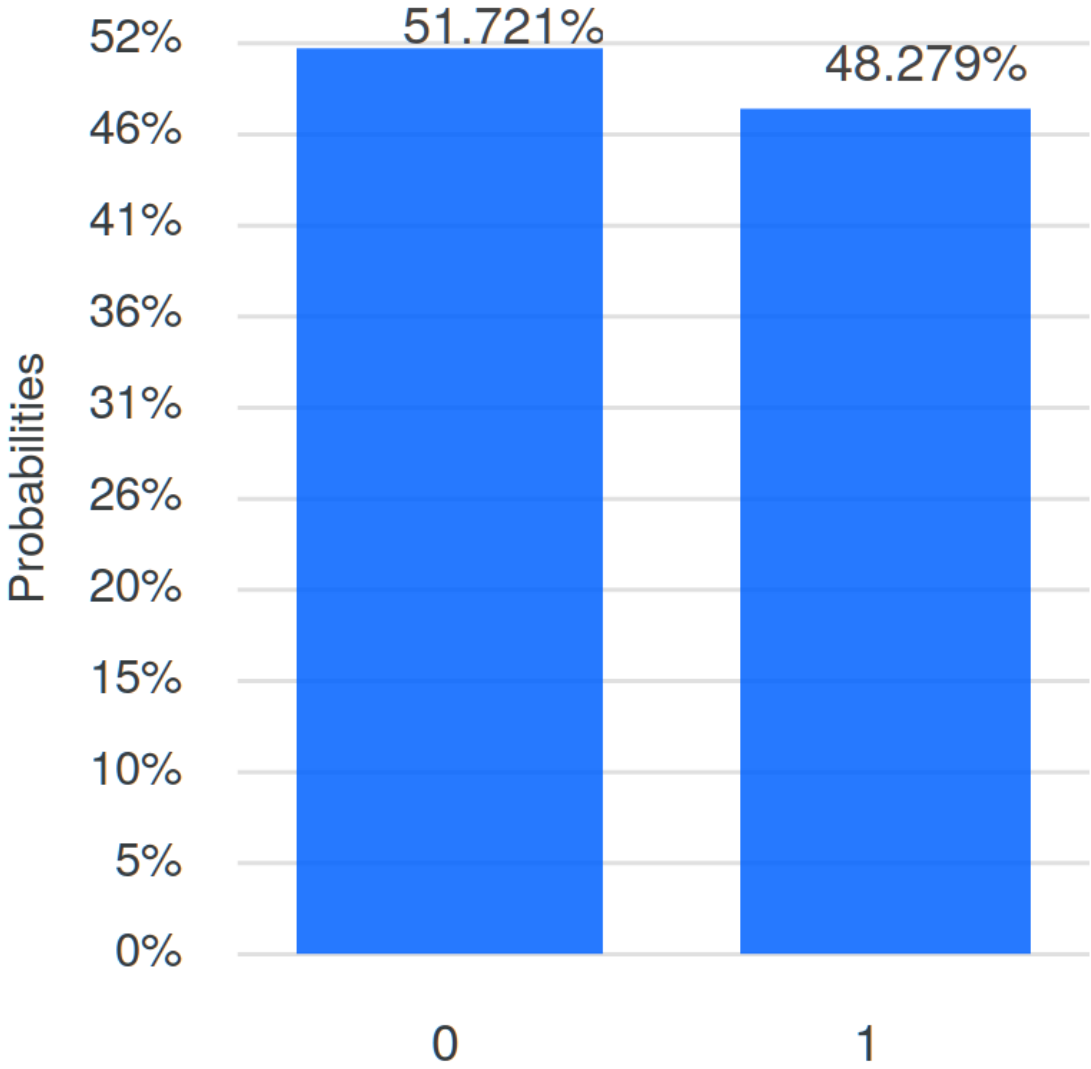






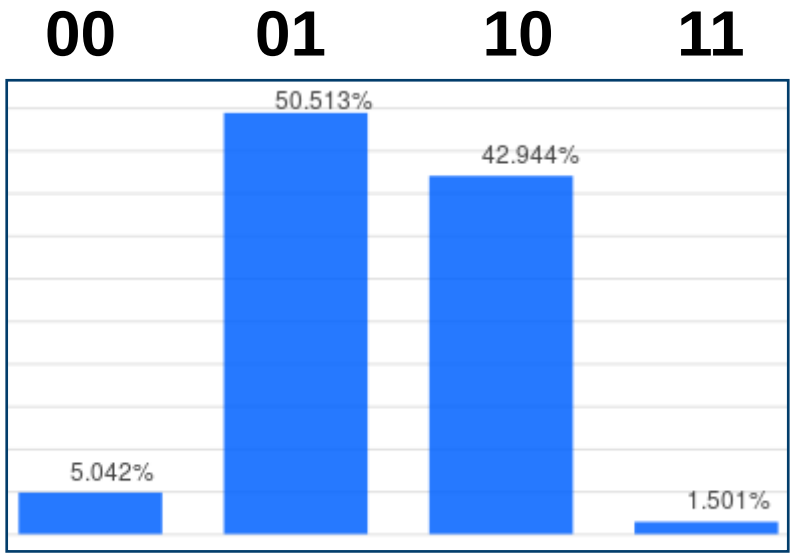


8192 Münzen:



GATTERBASIERTE QC: ANWENDUNGEN

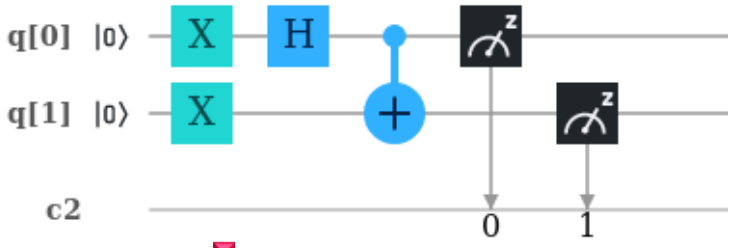
Münzwurf: 2 faire (?) Münzen



Was ist hier passiert?

→Geheime Absprachen!

“Entanglement”/Verschränkung



Enthält die geheimen Absprachen: $\frac{|01\rangle - |10\rangle}{\sqrt{2}}$

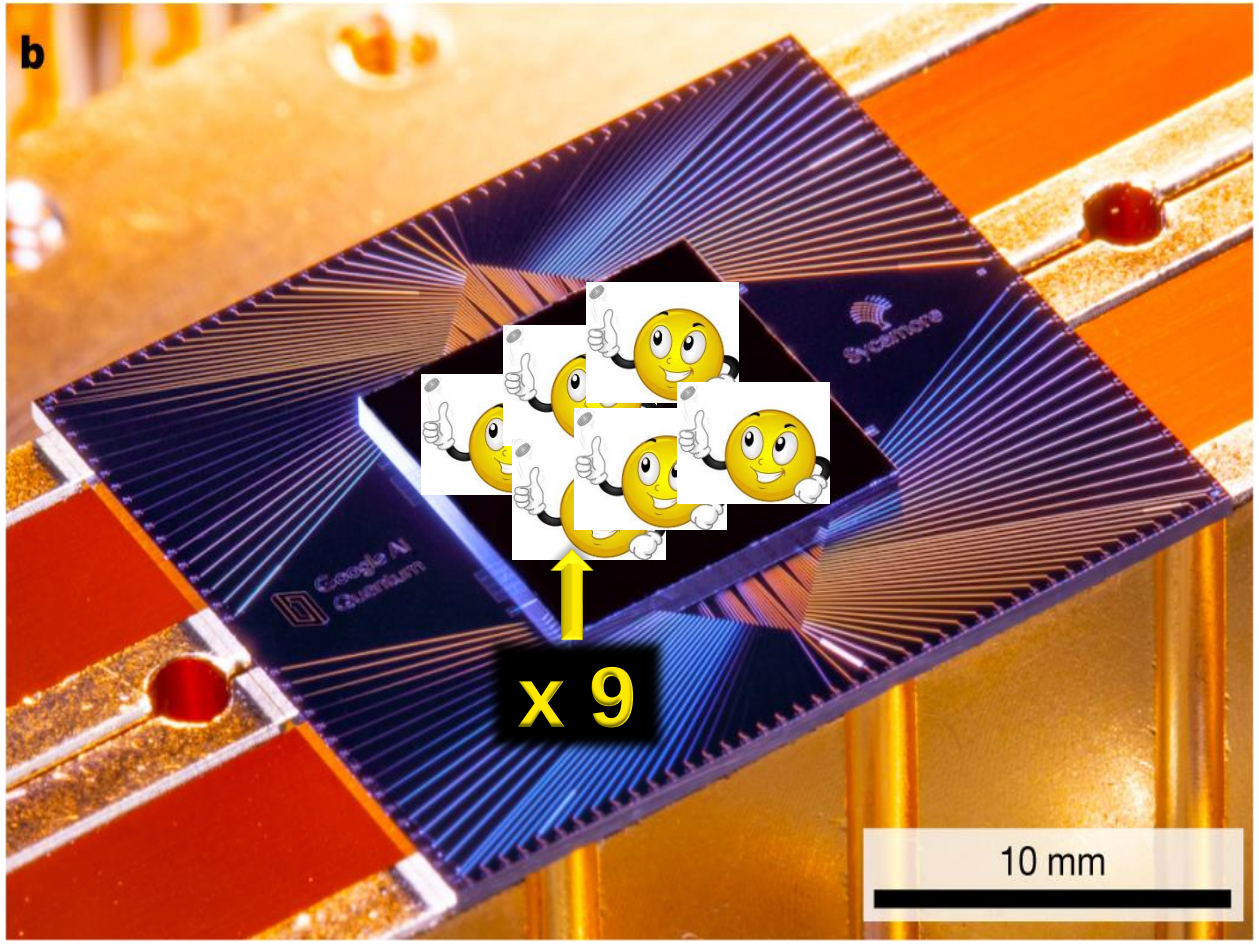
Woher kommen die Fehler?

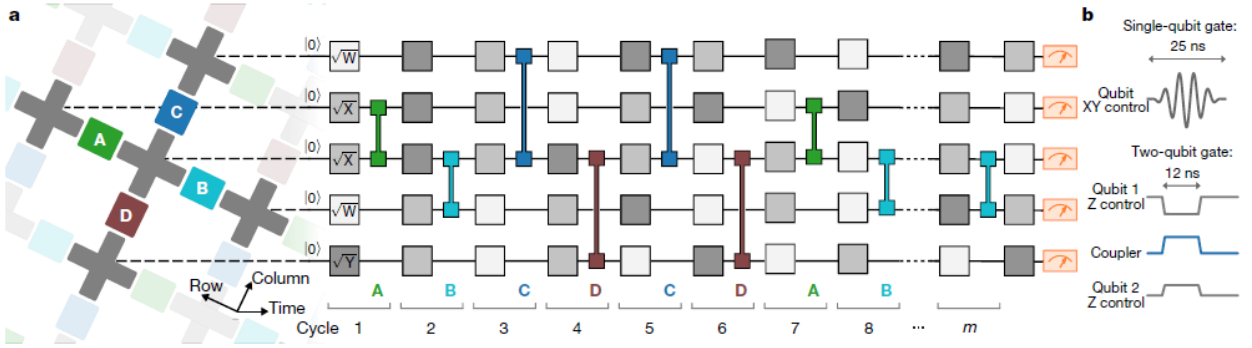
- Sie ignorieren die Absprache (Bit Flip Error)
- Sie vergessen die Absprache (Dekohärenz)
- Sie verlieren die Münze (Dissipation)

GATTERBASIERTE QC: ANWENDUNGEN

Arute et al., Nature 574, 505 (2019)
De Raedt et al., CPC 237, 41 (2019)
https://www.youtube.com/watch?v=JUSEYCkh2ac

Quantum Supremacy: 53 unfaire Münzen





Wie lauten die geheimen Absprachen?

→Bis zu **43 Qubits**:
Berechenbar mit

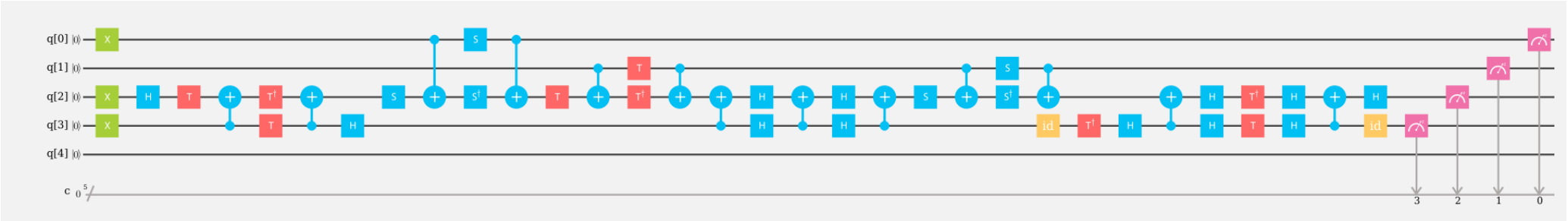
JUQCS (Jülich Universal Quantum Computer Simulator)

→Bis zu **45/48 Qubits**:
Berechenbar mit **JUQCS** (in Asien)

→Für **53 Qubits**? Unmöglich!
"Quantum Supremacy"

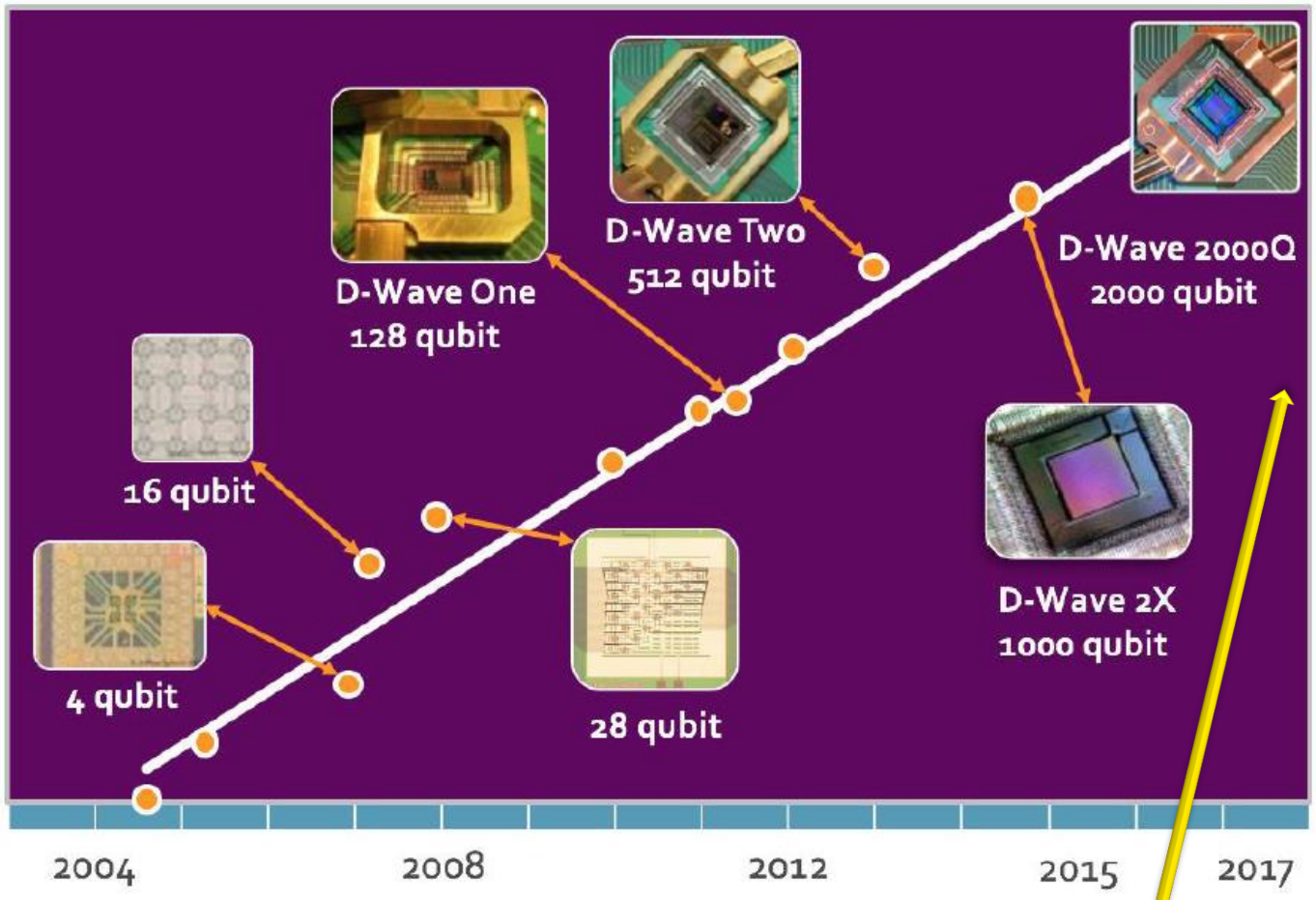
GATTERBASIERTE QC: ZUKÜNFTIGE ANWENDUNGEN

- **Shor:** Faktorisierung in Primzahlen
  - **Wichtig für:** Verschlüsselungen und andere sicherheitsrelevante Bereiche
  - **Problem:** Für $n = p * q$ finde Primfaktoren p und q (digitale Computer: praktisch unmöglich ab 1024 bit)
  - **Speedup durch Quantencomputer:** exponentiell O(log(n))
  - **Warum?** Quantenparallelismus: $|0\cdots000\rangle + |0\cdots001\rangle + |0\cdots010\rangle + |0\cdots011\rangle + \cdots + |1\cdots111\rangle$
- **Grover:** Suche in Datenbanken
  - **Problem:** In m Elementen $x_1, \ldots, x_m$ finde Element $\overline{x}$ (digitale Computer: O(m))
  - **Speedup durch Quantencomputer:** quadratisch O(sqrt(m))
- Prinzipiell: Alle digitalen Algorithmen, aber nicht unbedingt schneller → Beispiel: 2-bit Addition



QUANTENCOMPUTER #2

D-Wave Quantenannealer



Eigenschaften:

→Besitzt auch Münzwerfer (qubits)

→mit geheimen Absprachen (entanglement)

→aber einfacher zu steuern

QUANTENCOMPUTER #2

D-Wave Quantum Annealer

…einfacher – Was heißt das?

Quadratic Unconstrained Binary Optimization

“bias” (reelle Zahl)

$$\textbf{QUBO:} \quad \min_{q_i=0,1} \left(\sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j \right)$$

“coupler” / Koppler (reelle Zahl: enthält die geheimen Absprachen)

Warum interessant für **Anwender**?

→Diskrete Optimierung ist schwer (NP-schwer)

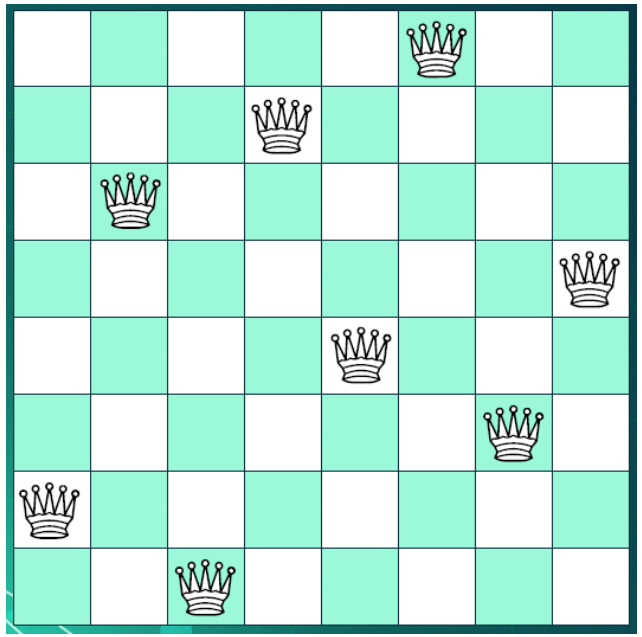
→Erhalte viele Lösungen **gleichzeitig**

→Sehr niedriger Energieverbrauch

IST ES SCHWER EINEN QUANTENANNEALER ZU NUTZEN?

Drei Schüler vom Gymnasium der Regensburger Domspatzen

N Queens Problem:
Gelöst für N = 8 auf dem DW2000Q



```
from dwave.system
    import DWaveSampler

a0 = -1
a4 = -1
b04 = 2

qubo = {(0,0): a0, (4,4): a4, (0,4): b04}

sampler = DWaveSampler()

response = sampler.sample_qubo(qubo,
    num_reads=1000)

print(response)
```

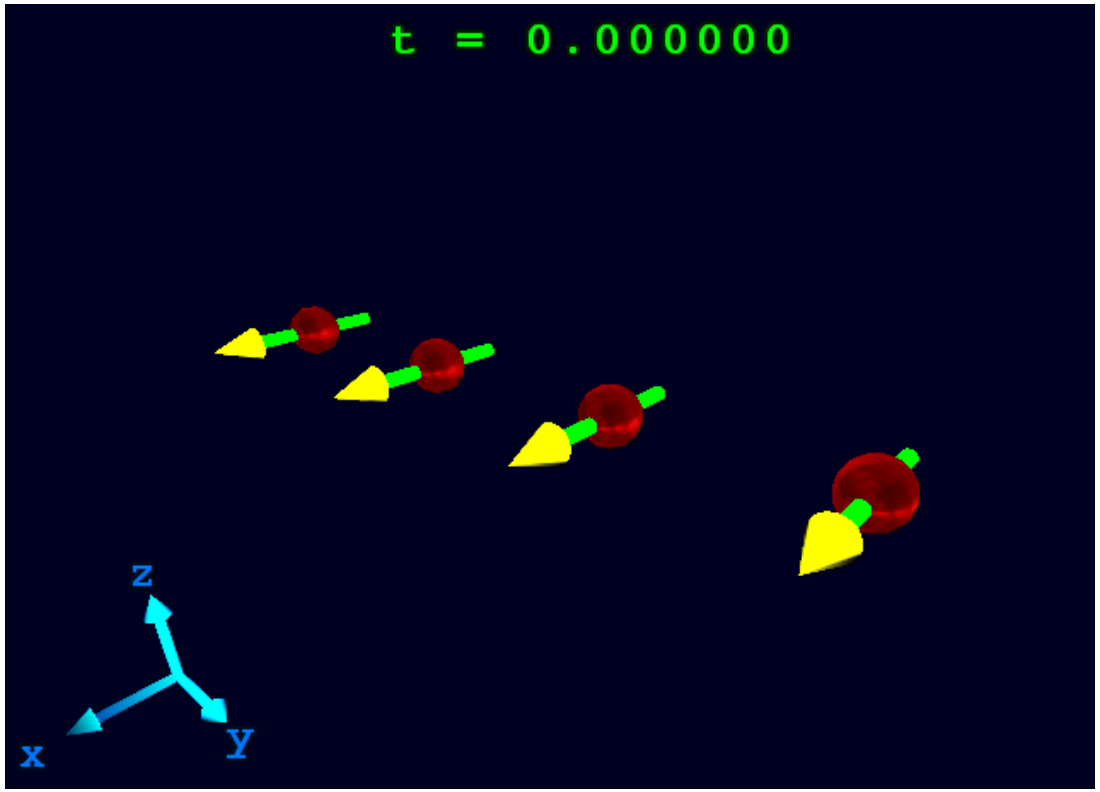
$$\sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j$$

QUANTENANNEALER

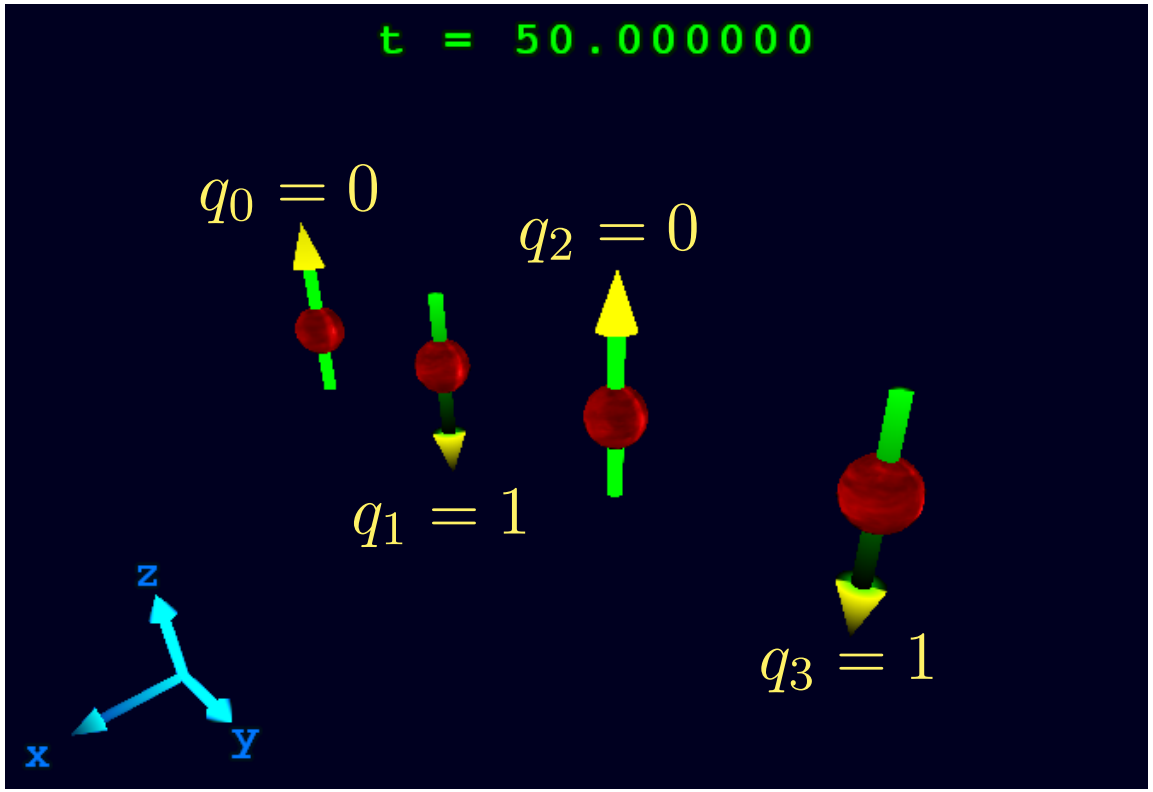
Wie funktioniert das?

$$\text{QUBO}: \quad \min_{q_0 q_1 q_2 q_3} \left(\sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j \right) \quad = \quad ????$$

Anfang der Berechnung



Ende der Berechnung

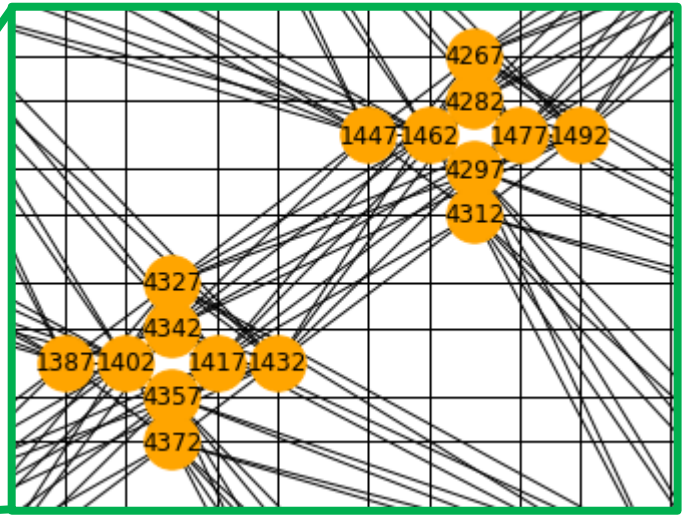


Quantum Annealing

QUANTENANNEALER: PROGRAMMIERUNG

$$\min_{q_i=0,1} \left(\sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j \right)$$

Eingabe: Programm
= Werte für a_i und b_{ij}



Architecture	DW2000Q	Advantage
Topology	Chimera	Pegasus
Nr. of qubits	~2000	~5000
Nr. of couplers	~6000	~35000
Max. connectivity	**6**	**15**

```
import dwave
from dwave.system import DWaveSampler
qubo = {
    (1387,1387): -1,
    (4327,4327): -0.5,
    (1387,4327): 2,
}
sampler =
DWaveSampler(solver='Advantage_system5.1')
results = sampler.sample_qubo(qubo, num_reads=100)
```

a_{1387}

a_{4327}

$b_{1387,4327}$

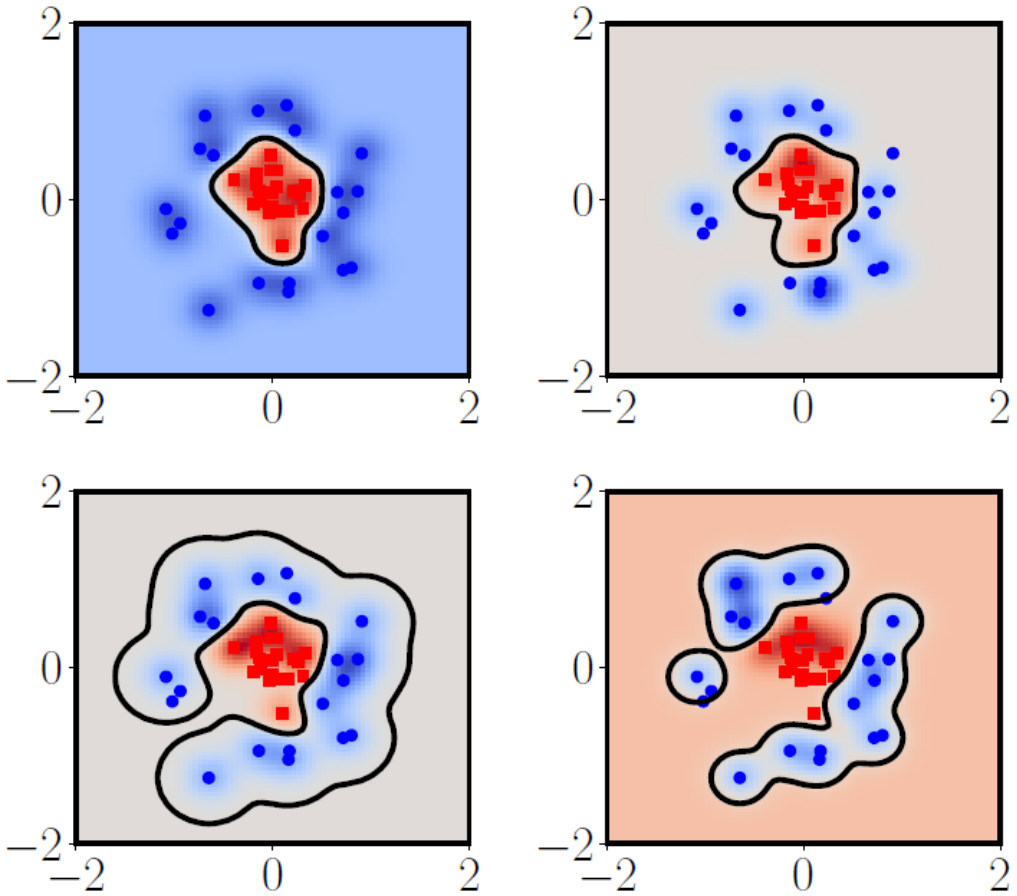
Ausgabe: Bitstrings

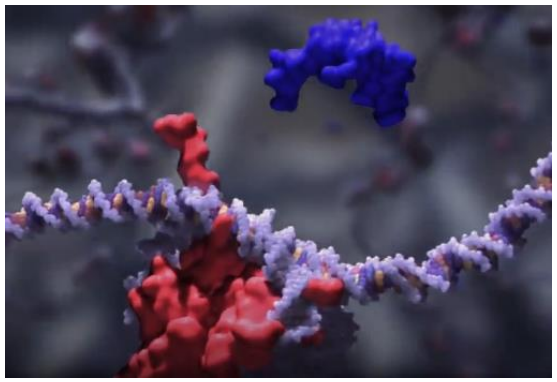
1387	4327	energy	num_occurrences
0	1	-0.5	90
0	0	0.0	5
1	0	0.0	5

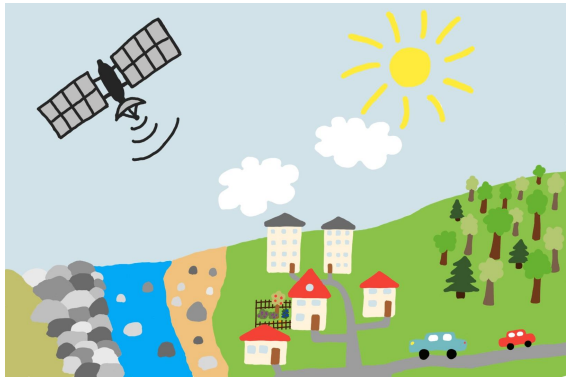
Willsch et al., CPC 248, 107006 (2020)
https://www.youtube.com/watch?v=mIgm67cUzrs

QUANTENANNEALER: ANWENDUNGEN

Klassifizierung (Support Vector Machines)







Tail Assignment:

Garden Optimization:

Unsere Beobachtungen:

☑ Vielzahl potentieller Lösungen **gleichzeitig**

☒ Problemgröße noch klein

ANWENDUNG: POKÉMON

- Problem: “Optimales Pokémon Team” (Jedes Qubit ⇔ Pokémontyp)
- QUBO: Minimiere “Schwäche des Teams” unter 3 Bedingungen:
  - 6 Pokémon in einem Team (6 Qubits “1”, N-6 Qubits “0”)
  - Gegen jeden Pokémontyp: ≥ 1 resistent
  - Gegen jeden Pokémontyp: ≤ 2 schwach
- Gewinner:

Pokémon Type Chart — Generation 1

created by pokemondb.net

0 No effect (0%) ½ Not very effective (50%) Normal (100%) 2 Super-effective (200%)

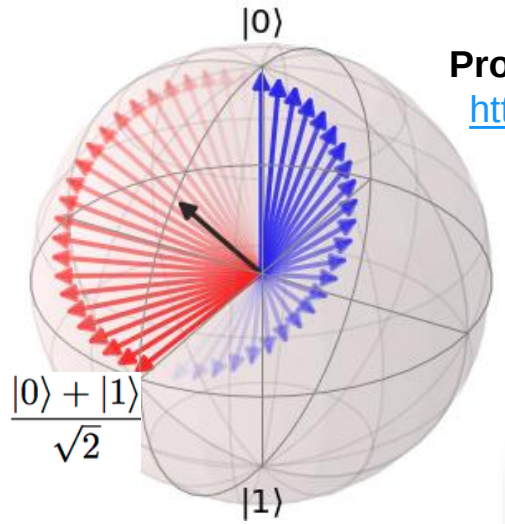
DEFENSE → ATTACK ↓	NOR	FIR	WAT	ELE	GRA	ICE	FIG	POI	GRO	FLY	PSY	BUG	ROC	GHO	DRA
NORMAL													½	0	
FIRE		½	½		2	2						2	½		½
WATER		2	½		½				2				2		½
ELECTRIC			2	½	½				0	2					½
GRASS		½	2		½			½	2	½		½	2		½
ICE			½		2	½			2	2					2
FIGHTING	2					2		½		½	½	½	2	0	

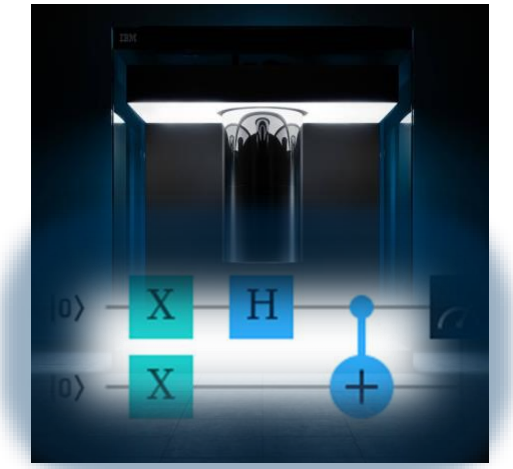
ZUSAMMENFASSUNG

Lecture Notes:
Programming Quantum Computers
https://arxiv.org/pdf/2201.02051.pdf

- **Typen: Gatterbasierte QC & Quantenannealer**
- **Fundamentale Recheneinheit: 1 Qubit**

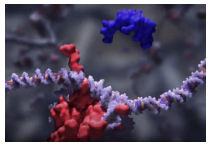
  “Münzwerfer” mit Absprachen
- **Momentaner Entwicklungsstand:**

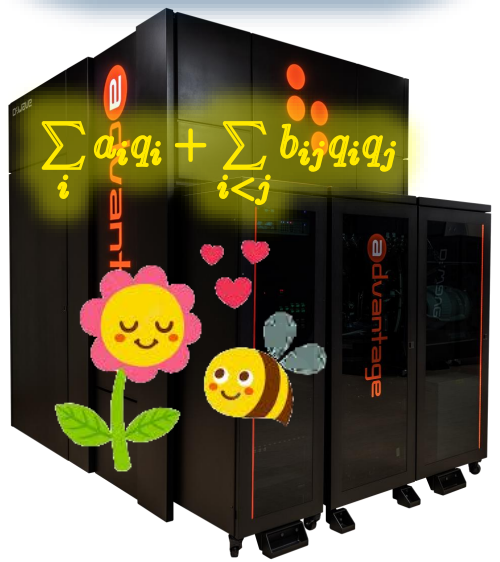




Industrielle Anwendungen

Gatterbasierte QC **Quantenannealer**





- **Programmierung in Python: IBM Qiskit, D-Wave Ocean SDK, Google Cirq, ...**
- **Werden digitale Computer durch Quantencomputer ersetzt? Nein**
- **Nutzermodell: Programme schicken spezielle Teilprobleme an Quantencomputer**
- **Wofür relevant? → Faktorisierung, Suche in Datenbanken, Optimierung, ...**
- **→ Spannende Technologie mit unabsehbarer Zukunft**

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT

- Weitere Informationen:
  - **Programming Quantum Computers:** https://arxiv.org/pdf/2201.02051.pdf
  - **JUQCS:** Simulation of Quantum Computers on GPUs https://www.youtube.com/watch?v=JUSEYCkh2ac
  - **Applications on Quantum Annealers at JSC:** https://www.youtube.com/watch?v=mIgm67cUzrs
  - **The Quantum Computer Building at JSC:** https://go.fzj.de/juniq-tour
  - **JUNIQ:** Quantum Computing in Your Browser https://juniq.fz-juelich.de