Quantum Simulation of Physical Systems

Cameron Cogburn

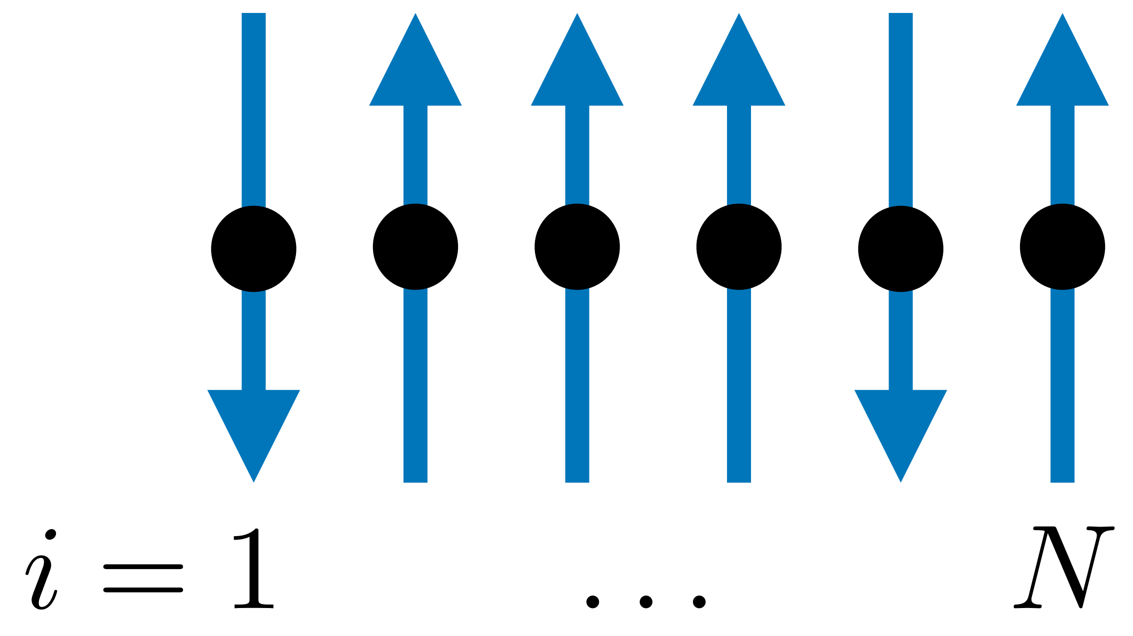
Quantum Club Oct. 23, 2024

Quantum Simulation

- As scientists, a primary motivation is to understand the world at a fundamental level.

- Because the world is inherently quantum, we need an inherently quantum device to accurately simulate physical systems.

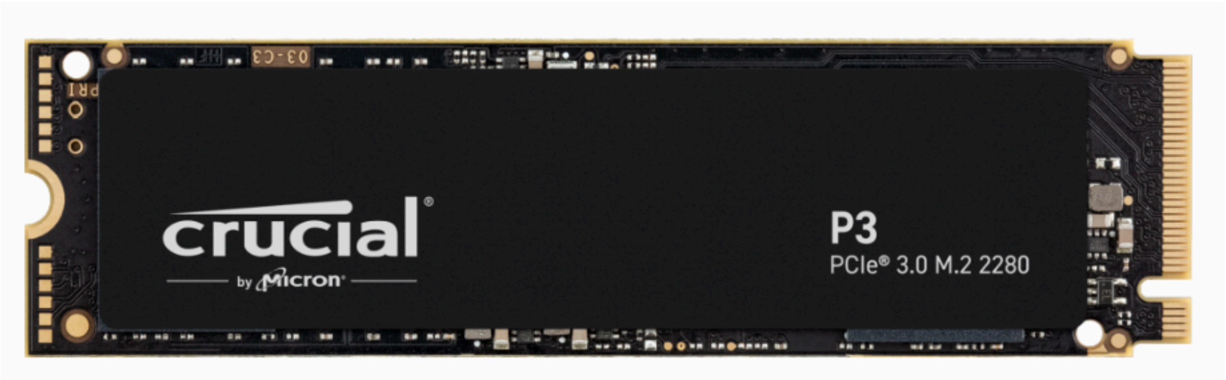
- One reason this is the case is because the number of possible configurations of a system (Hilbert space) grows exponentially in size $\mathcal{O}(e^N)$.



- 2^N unique configurations. When $N \sim 300$ the volume of the memory needed to store this would exceed the volume of the observable universe…

Simulating Dynamics

Why can't we just use a powerful enough computer? Take a 4 TB SSD chip:



$$V_{\text{chip}} = 80.01 \times 22.1 \times 2.29 \text{ mm}^3$$

How many spins could the universe hold in classical memory? $V_{\text{univ}} = 3.57 \cdot 10^{80} \text{ m}^3$

$$\left(\frac{4 \text{ TB}}{V_{\text{chip}}}\right) V_{\text{univ}} = (32 \cdot 2^N \text{ bits}) \left(\frac{1 \text{ TB}}{8 \cdot 10^{12} \text{ bits}}\right)$$

$$\implies \quad N \approx 325$$

Hamiltonian Dynamics

- Quantum time evolution of an isolate state given by the Schrödinger Equation:

$$i\frac{d}{dt}|\phi(t)\rangle = \hat{H}_{\mathrm{s}}(t)|\phi(t)\rangle$$

- Formal solution:

$$|\phi(t)\rangle = T\exp\left(-i\int_0^t ds\,\hat{H}_{\mathrm{s}}(s)\right)|\phi(0)\rangle$$

- For a time-independent Hamiltonian:

$$|\phi(t)\rangle = e^{-it\hat{H}_{\mathrm{s}}}|\phi(0)\rangle$$

Simulating Dynamics

To simulate Hamiltonian dynamics on a quantum computer we need to:

- Map (exactly or approximately) the state $|\phi(t)\rangle$ to a state on the quantum computer

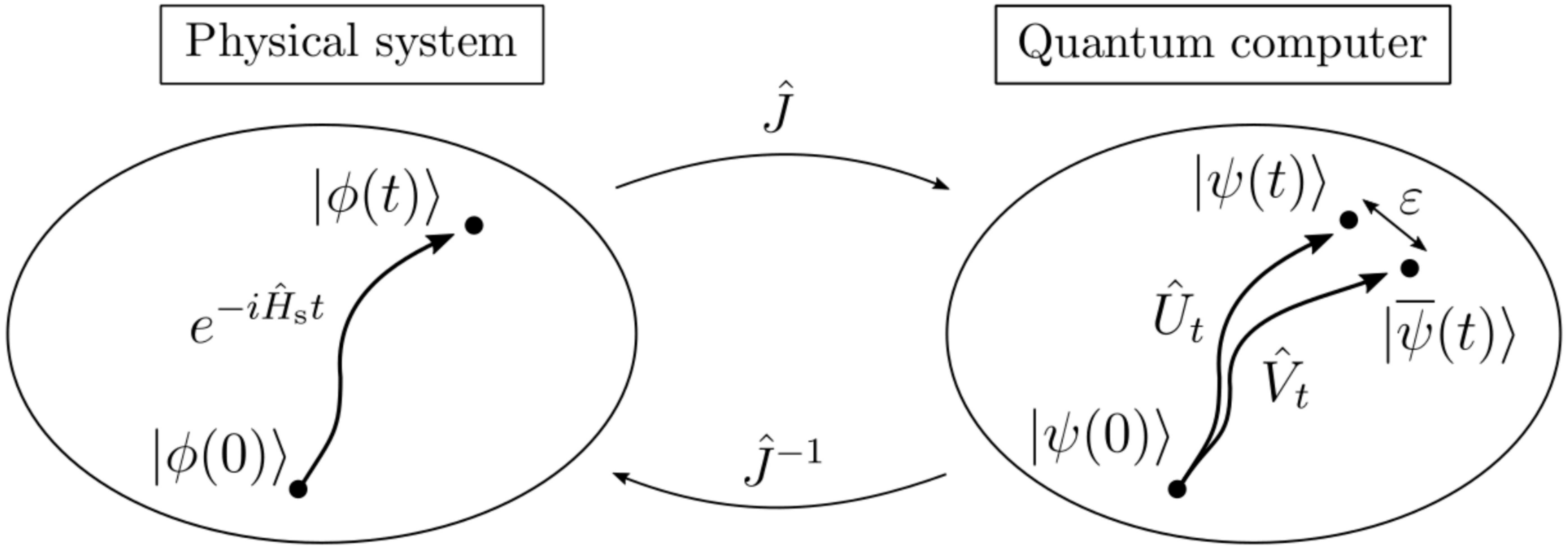
- Calculate or perform a measurement of a physical observable $\hat{O}$ at time t :

$$O(t) = \langle\phi(t)|\hat{O}|\phi(t)\rangle$$

Examples of observables: magnetization, correlations, charge, etc.

Digital Quantum Simulation

How do we simulate a physical system on quantum hardware?



$$\hat{U}_t = e^{-i\hat{H}t}$$

$$\hat{V}_t = \text{circuit}$$

$$|\phi(t)\rangle \xmapsto{\hat{J}} |\psi(t)\rangle = \hat{J}|\phi(t)\rangle$$

$$\hat{H}_s \xmapsto{\hat{J}} \hat{H} = \hat{J}\hat{H}_s\hat{J}^{-1}$$

Digital Quantum Simulation

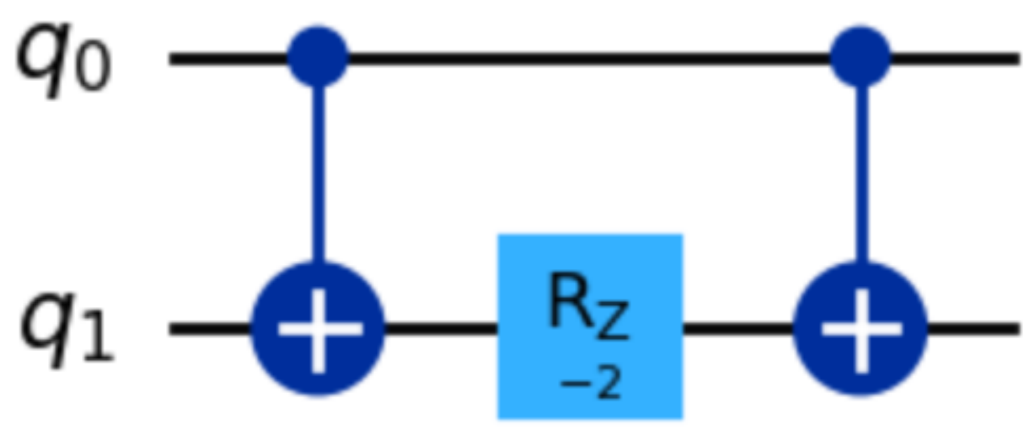
Four powerful properties about quantum simulations:

1. If $\hat{H}$ can be simulated efficiently, then so can $c\hat{H}$ with $c \in \mathbb{R}$.
2. If $\hat{H}_1$ and $\hat{H}_2$ can be simulated efficiently, then so can $\hat{H}_1 + \hat{H}_2$.
3. If $\hat{H}$ is diagonal, $\hat{H}$ can be simulated efficiently.
4. If $\hat{H}$ can be simulated efficiently and $\hat{U}$ is a unitary operator that can be implemented efficiently on quantum hardware, then $\hat{H}' = \hat{U}\hat{H}\hat{U}^{-1}$ can be simulated efficiently.

Digital Quantum Simulation

One final powerful property: the exponential map of Pauli matrices

$$e^{i\hat{\sigma}_z \otimes \hat{\sigma}_z} =$$



Proof. Use that $e^{i\hat{A}t} = \cos(t)I + i\sin(t)\hat{A}$ $\qquad \mathrm{CNOT} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes \hat{\sigma}_x$

to show that $\mathrm{CNOT}(I \otimes e^{i\hat{\sigma}_z t})\mathrm{CNOT} = e^{i\hat{\sigma}_z \otimes \hat{\sigma}_z t}$

Trotterization

If the terms of the Hamiltonian commute, $[\hat{H}_i, \hat{H}_j] = 0$, then time evolution given by

$$\hat{U}_t = e^{-i\hat{H}t} = e^{-i(\hat{H}_1 + \cdots + \hat{H}_L)t} = e^{-i\hat{H}_1 t} \cdots e^{-i\hat{H}_L t}$$

If the terms of the Hamiltonian do not commute, $[\hat{H}_i, \hat{H}_j] \neq 0$, then

$$\hat{U}_{\Delta t} = e^{-i\hat{H}\Delta t} \approx e^{-i\hat{H}_1 \Delta t} \cdots e^{-i\hat{H}_L \Delta t} \equiv \hat{V}_{\Delta t}$$

Quantify error by using expanding both sides and using Baker-Campbell-Hausdorff.

$$e^A e^B = e^C \,, \quad C = A + B + \frac{1}{2}[A, B] + \frac{1}{12}[A, [A, B]] + \cdots$$

How far you expand to determines the order of the Trotter expansion.

Digital Quantum Simulation

Substituting in
$$\hat{H}_{\text{s}} = \omega(\hat{A}_x \otimes \hat{A}_y \otimes \hat{A}_z)(\hat{\sigma}_x \otimes \hat{\sigma}_y \otimes \hat{\sigma}_z)(\hat{A}_x^\dagger \otimes \hat{A}_y^\dagger \otimes \hat{A}_z^\dagger)$$
$$= \omega \hat{A}(\hat{\sigma}_x \otimes \hat{\sigma}_y \otimes \hat{\sigma}_z)\hat{A}^\dagger$$

Additionally,
$$\hat{\sigma}_x \otimes \hat{\sigma}_y \otimes \hat{\sigma}_z = c_2\hat{X}_1 c_1 \hat{X}_0 (\mathbb{1} \otimes \mathbb{1} \otimes \hat{\sigma}_z) c_1 \hat{X}_0 c_2 \hat{X}_1 = \hat{B}(\mathbb{1} \otimes \mathbb{1} \otimes \hat{\sigma}_z)\hat{B}^\dagger$$

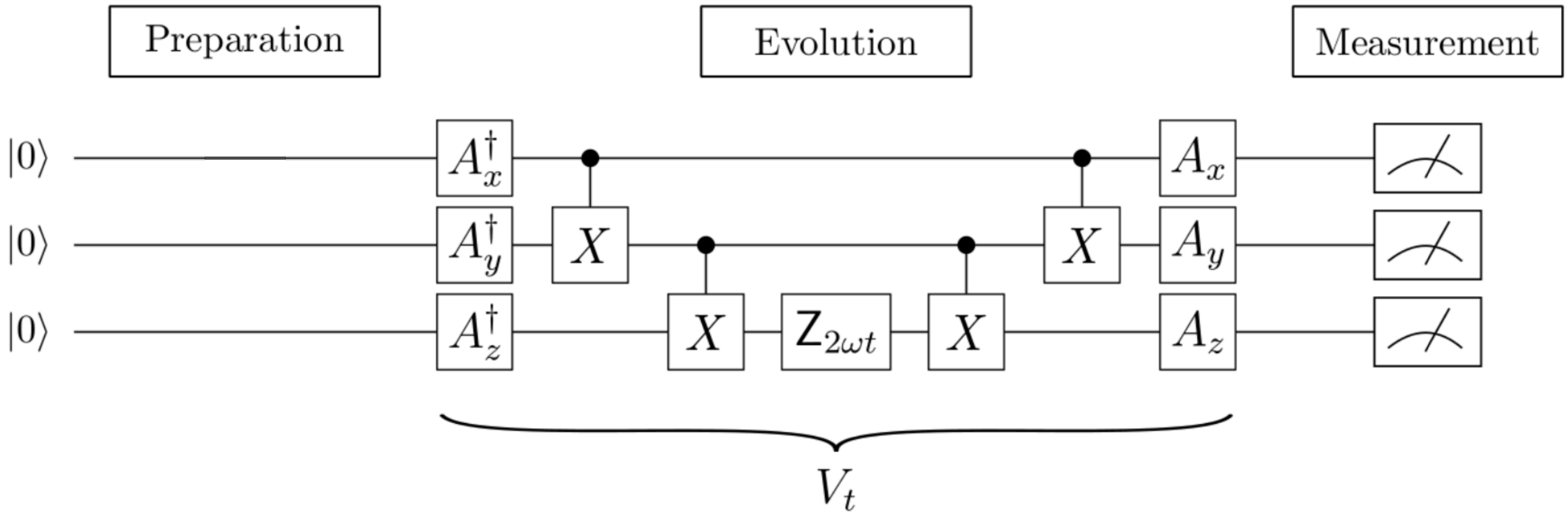
The Hamiltonian is then
$$\hat{H} = \omega \hat{A}\hat{B}(\mathbb{1} \otimes \mathbb{1} \otimes \hat{\sigma}_z)\hat{B}^\dagger \hat{A}^\dagger$$

Time evolution operator
$$\hat{U}_t = e^{-i\hat{H}t} = e^{-i\omega \hat{A}\hat{B}(\mathbb{1}\otimes\mathbb{1}\otimes\hat{\sigma}_z)\hat{B}^\dagger \hat{A}^\dagger} = \hat{A}\hat{B}e^{-i\omega t(\mathbb{1}\otimes\mathbb{1}\otimes\hat{\sigma}_z)}\hat{B}^\dagger \hat{A}^\dagger$$

The quantum circuit is therefore
$$\hat{V}_t = \hat{A}\hat{B}(\mathbb{1} \otimes \mathbb{1} \otimes \mathsf{Z}_{2\omega t})\hat{B}^\dagger \hat{A}^\dagger$$

Digital Quantum Simulation

The quantum circuit is therefore $\hat{V}_t = \hat{A}\hat{B}(\mathbb{1} \otimes \mathbb{1} \otimes \mathsf{Z}_{2\omega t})\hat{B}^\dagger \hat{A}^\dagger$



Digital Quantum Simulation

Let's do this on Qiskit!

We will:

- Construct the circuit 'by hand' and using PauliEvolutionGate.
- Observe the average magnetization as a function of time.
- Compare the results.

References

- Mario Motta’s Hamiltonian Dynamics talk on YouTube: https://youtu.be/FN2BGJOYqKo?si=sY6DOeNZ_A6MY_Aq

- Quantum Information Science by Riccardo Manenti and Mario Motta.

Spin-Boson Model

Spin-Boson Model

- A spin-boson model is a system of interacting spins and bosons, which can be used to model a variety of physical systems.

- Useful to describe:

    - Quantum impurities: single spin, many bosons

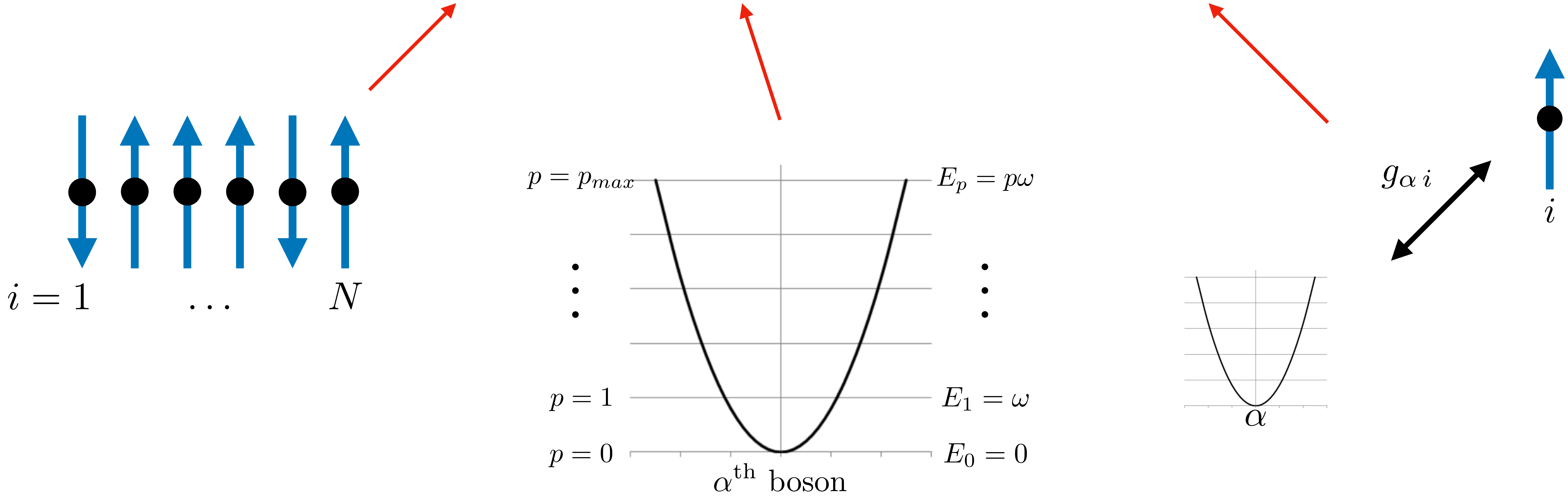
    - Superradiance: single boson (cavity mode), many spins

    - Noise in QCs! (Entanglement between a qubit and its environment)

Spin-Boson Model

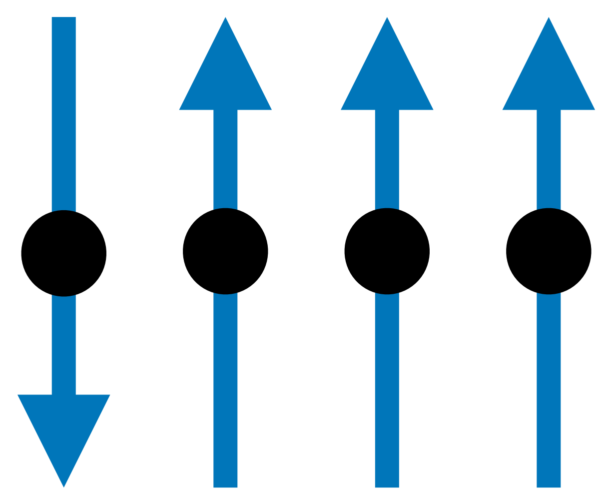
- The spin-boson model is a system of *N* interacting spins and *M* bosons, which have *p* occupation levels. This can be used to model a variety of physical systems.

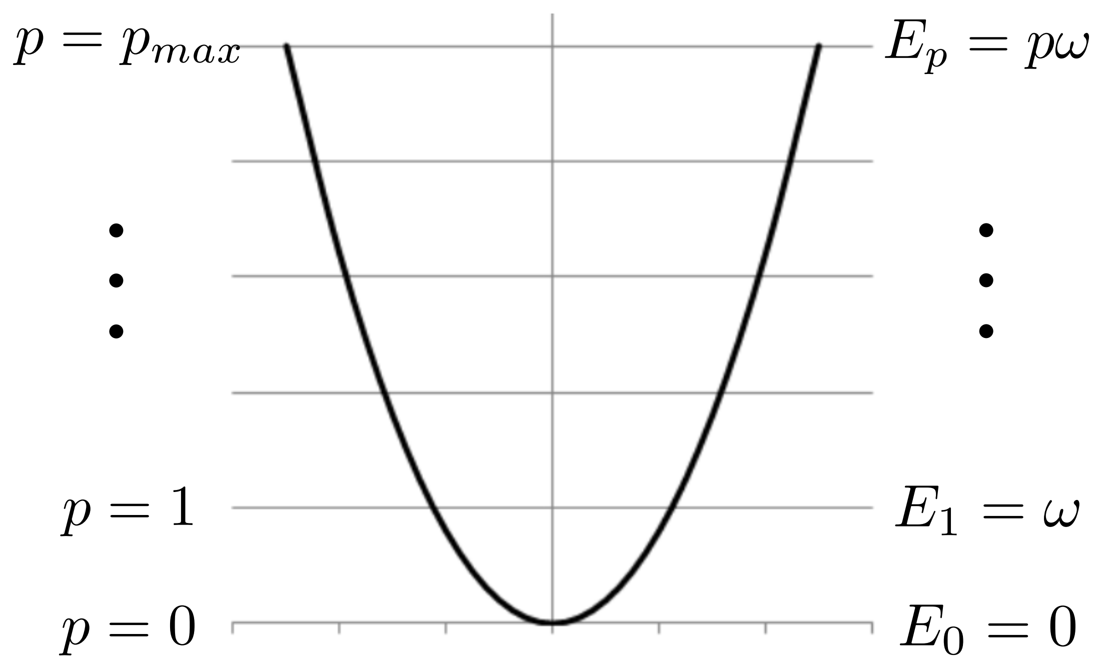
$$H_{N,M,p} = \frac{\Delta}{2}\sum_{i=1}^{N}\sigma_i^z + \sum_{\alpha=1}^{M}\omega_\alpha a_\alpha^\dagger a_\alpha + \frac{1}{\sqrt{N}}\sum_{\alpha\, i}\frac{g_{\alpha\, i}}{2}(a_\alpha + a_\alpha^\dagger)\sigma_i^x$$



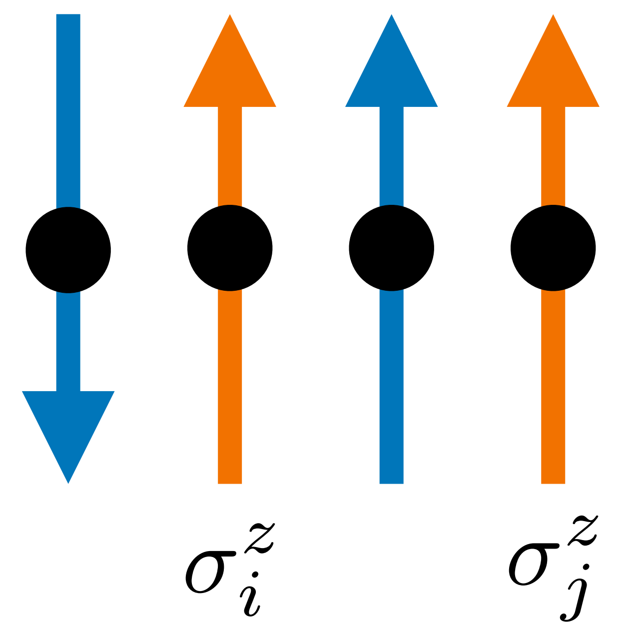
Observables

- Three natural observables:
    - Average magnetization: $M(t) = \frac{1}{N}\sum_{i=1}^{N}\langle\sigma_i^z(t)\rangle$





    - Occupation number: $N(t) = \langle a^\dagger(t)a(t)\rangle$
    - Spin correlator: $C_{ij}(t) = \langle\sigma_i^z(t)\sigma_j^z(t)\rangle$

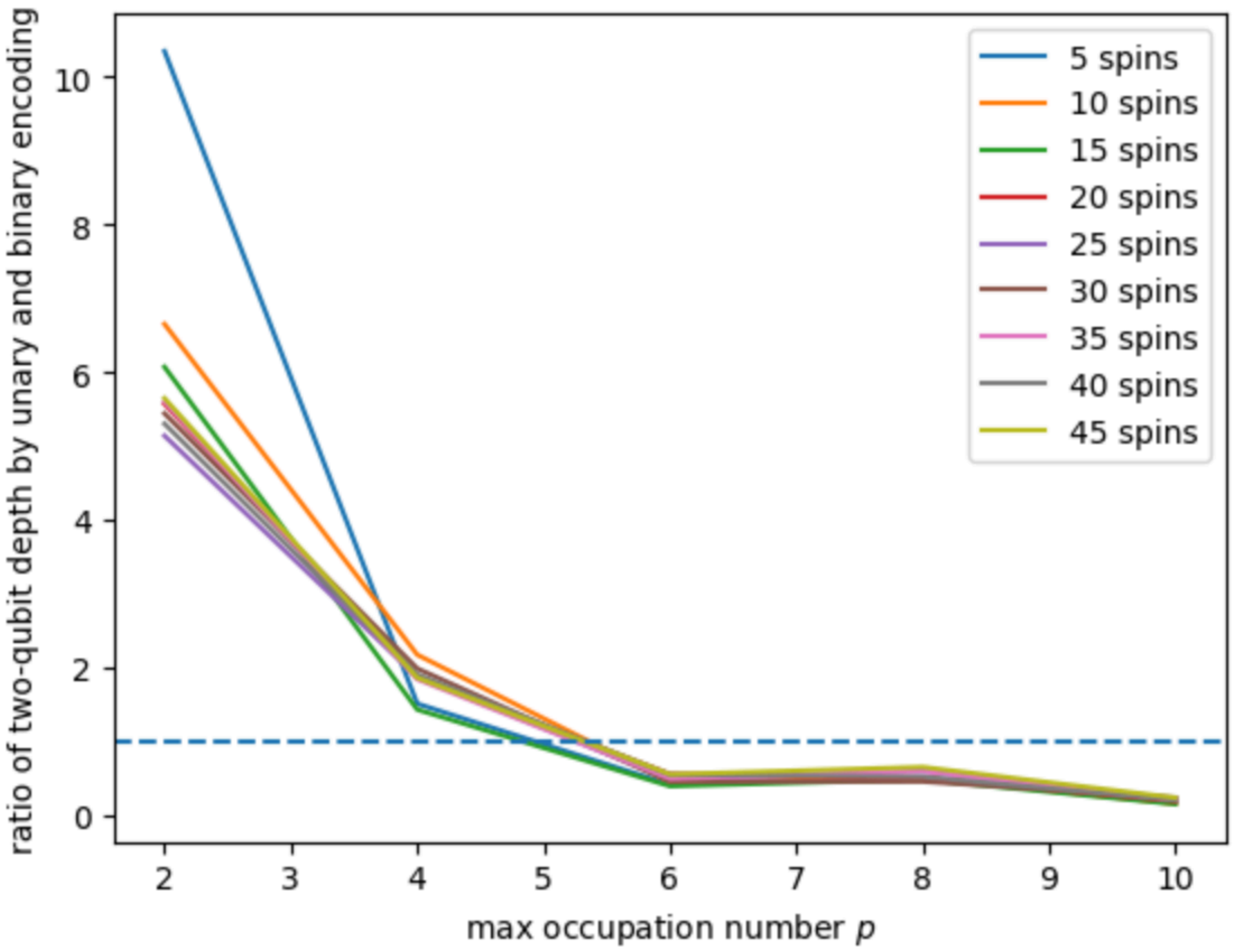


Mapping to Hardware

- How to encode an infinite number of boson modes? First, note that low-energy dynamics don't involve higher states, so truncation is acceptable.
- Suppose have *m* total boson modes (*m=Mp)*. There are two established strategies:
- Unary encoding: $|m\rangle \to |00\ldots1\ldots00\rangle$
- Binary encoding: $$|m\rangle \to \prod_i \left| \left\lfloor \frac{m}{2^i} \right\rfloor \mod 2 \right\rangle$$
- In qubit number, unary vs binary goes as $\mathcal{O}(m)\,\mathrm{vs}\,\mathcal{O}(\log m)$
- But in *gate* number, unary vs binary goes as $\mathcal{O}(m)\,\mathrm{vs}\,\mathcal{O}(m\log m)$

Unary vs binary encoding

- Can see 2-qubit depth of unary vs binary encoding schemes

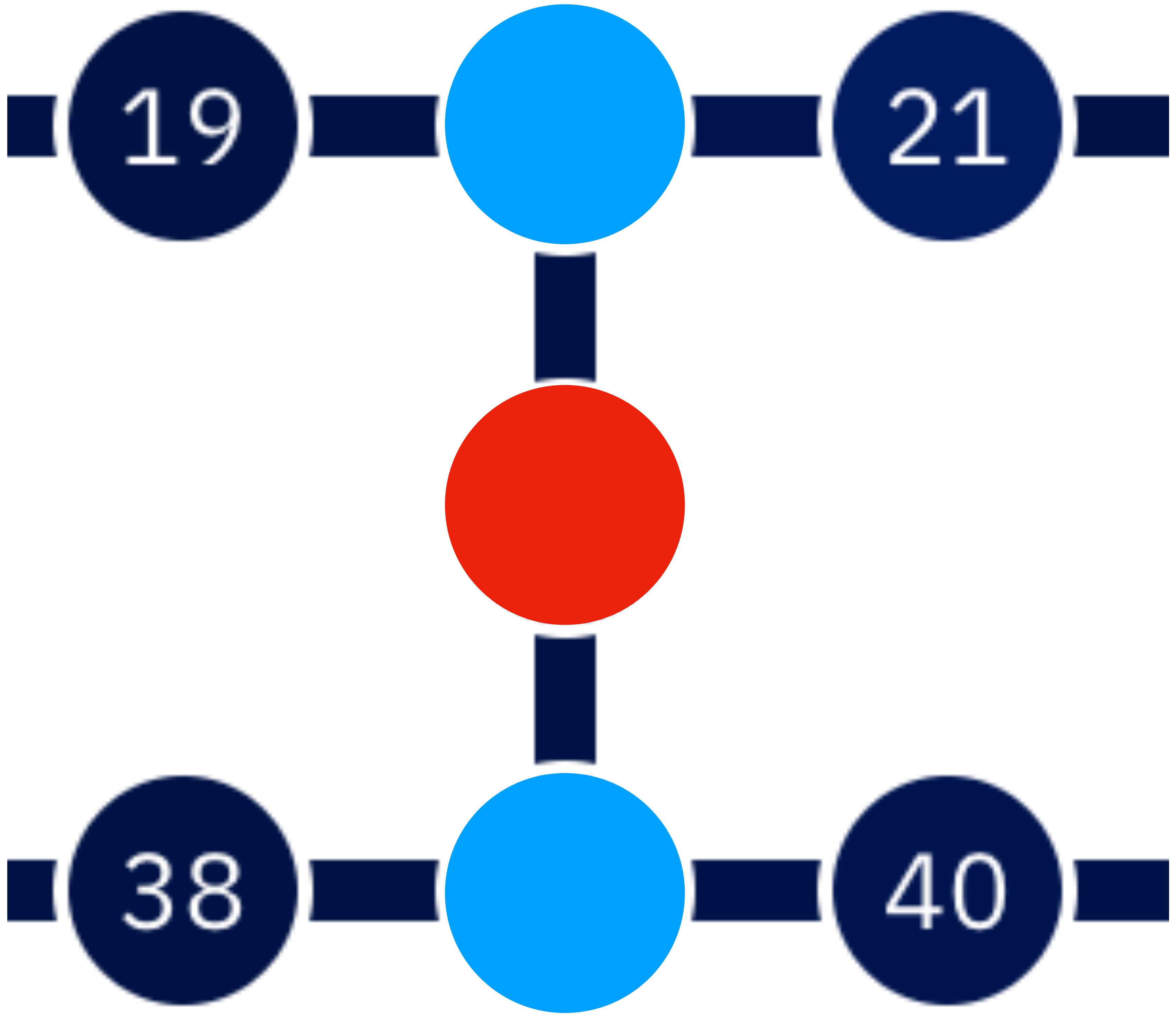


Mapping to Hardware

- First look at smallest non-trivial system using binary encoding:

$$N = 2$$

$$M = 1, p = 2$$



The Software Qiskit

- Qiskit has a number of built-in functions for straight-forward implementation!

$$H_{2,1,2} = \frac{\Delta}{2}(\sigma_1^z + \sigma_2^z) + \frac{\omega}{2}(1 - \tau^z) + \frac{g}{2\sqrt{2}}(\tau^x \sigma_1^x + \tau^x \sigma_2^x)$$

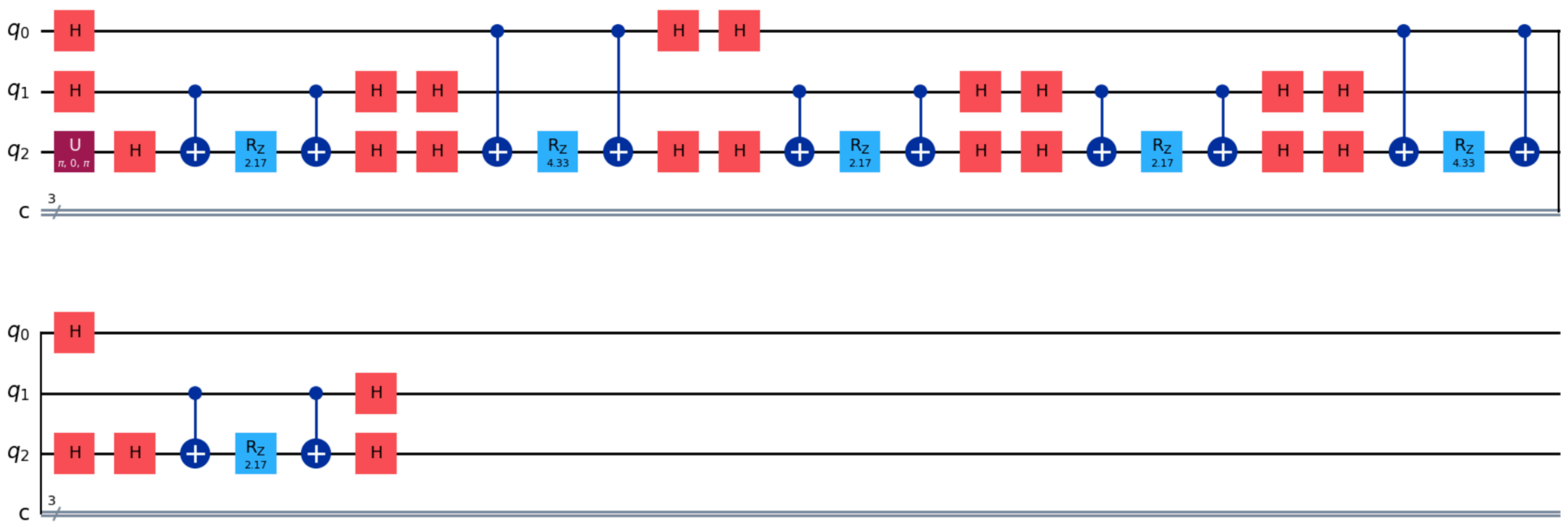
```
SBHam212 = SparsePauliOp.from_list([
                                    ("IZI", delta/2), ("IIZ", delta/2),
                                    ("III", omega/2), ("ZII", -omega/2) ,
                                    ("XXI", (g/(2*np.sqrt(2)))), ("XIX", (g/(2*np.sqrt(2))))])
N = SBHam212.num_qubits

observable_corr = SparsePauliOp.from_list([ ("IZZ", 1) ])
observable_mag = SparsePauliOp.from_list([ ("IZI", 1/2), ("IIZ", 1/2) ])
observable_num_occ = SparsePauliOp.from_list([ ("III", 1/2), ("ZII", -1/2) ])
```

$$C_{12}(t) = I \otimes \sigma^z(t) \otimes \sigma^z(t) \qquad M(t) = \frac{1}{2}(I \otimes \sigma^z(t) \otimes I + I \otimes I \otimes \sigma^z(t)) \qquad N(t) = \frac{1}{2}(I - \tau^z(t)) \otimes I \otimes I$$

The Software

```
circuit.decompose(reps=3).draw("mpl")
```




```
circuit.decompose(reps=3).depth()
```

35

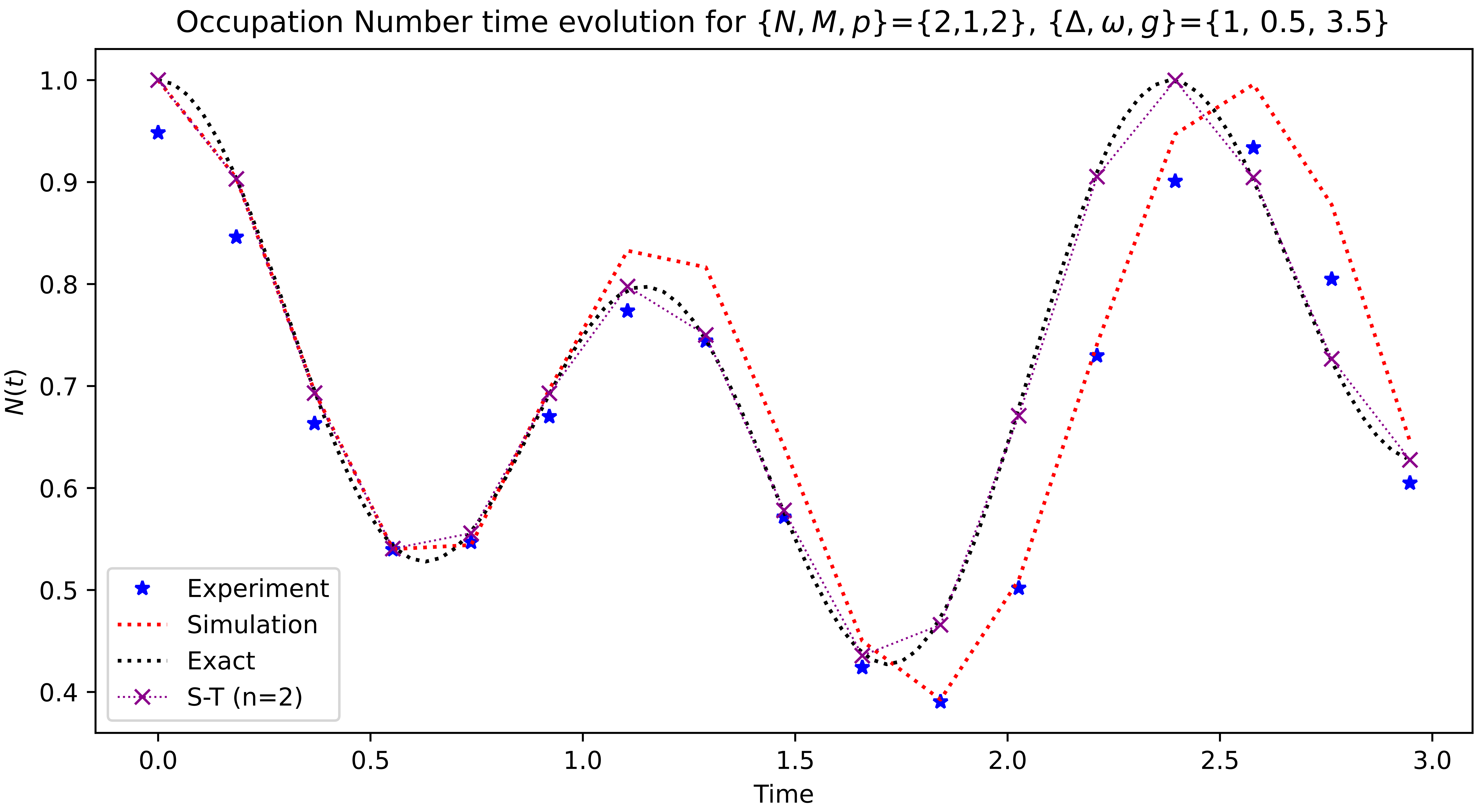
The Hardware

- IBM System One Eagle Processor with 127 qubits

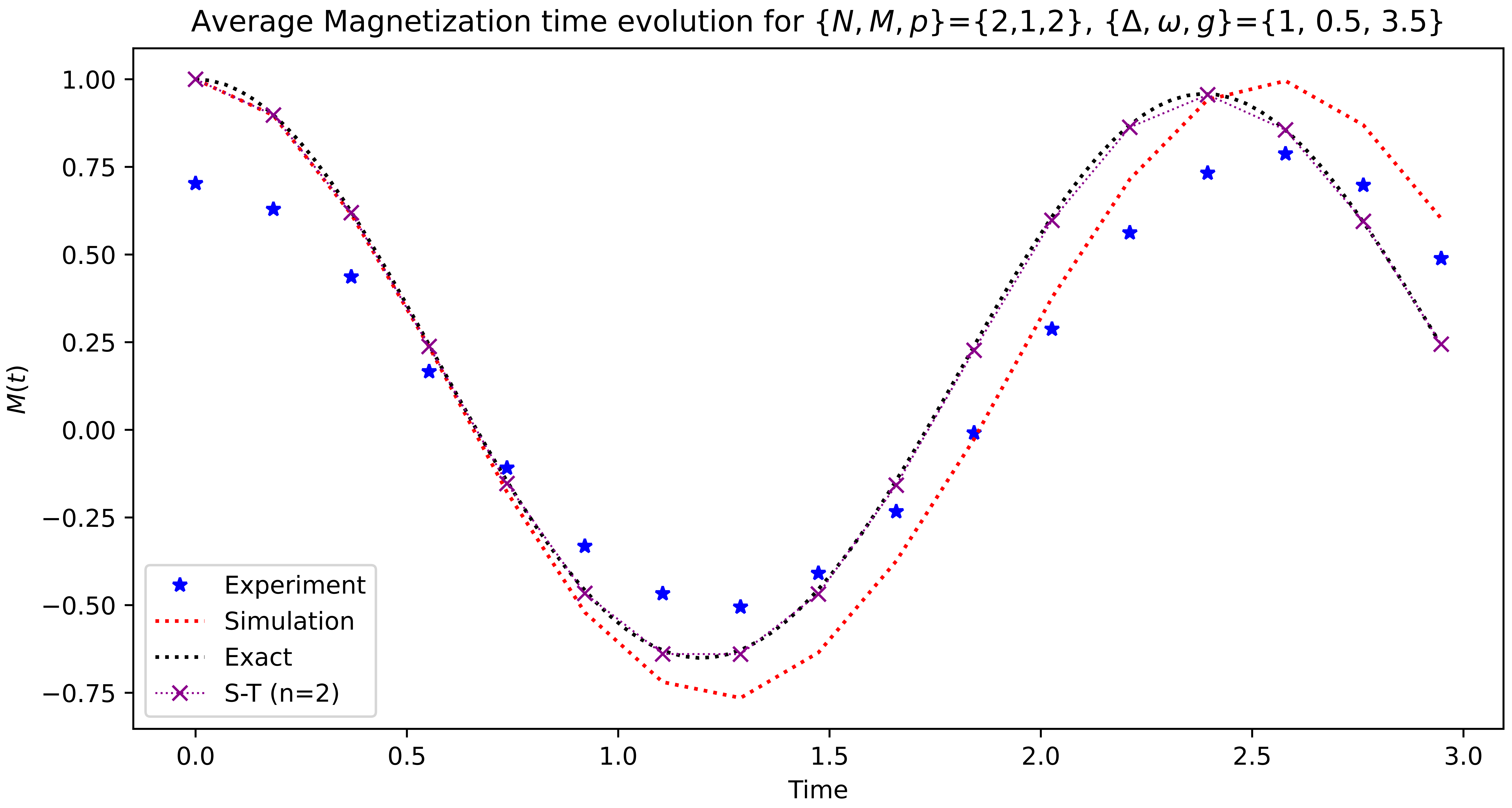
```
from qiskit_ibm_runtime import QiskitRuntimeService

backend_name = "ibm_rensselaer"
service = QiskitRuntimeService(
    channel='ibm_quantum',
)
backend = service.get_backend(backend_name)
```

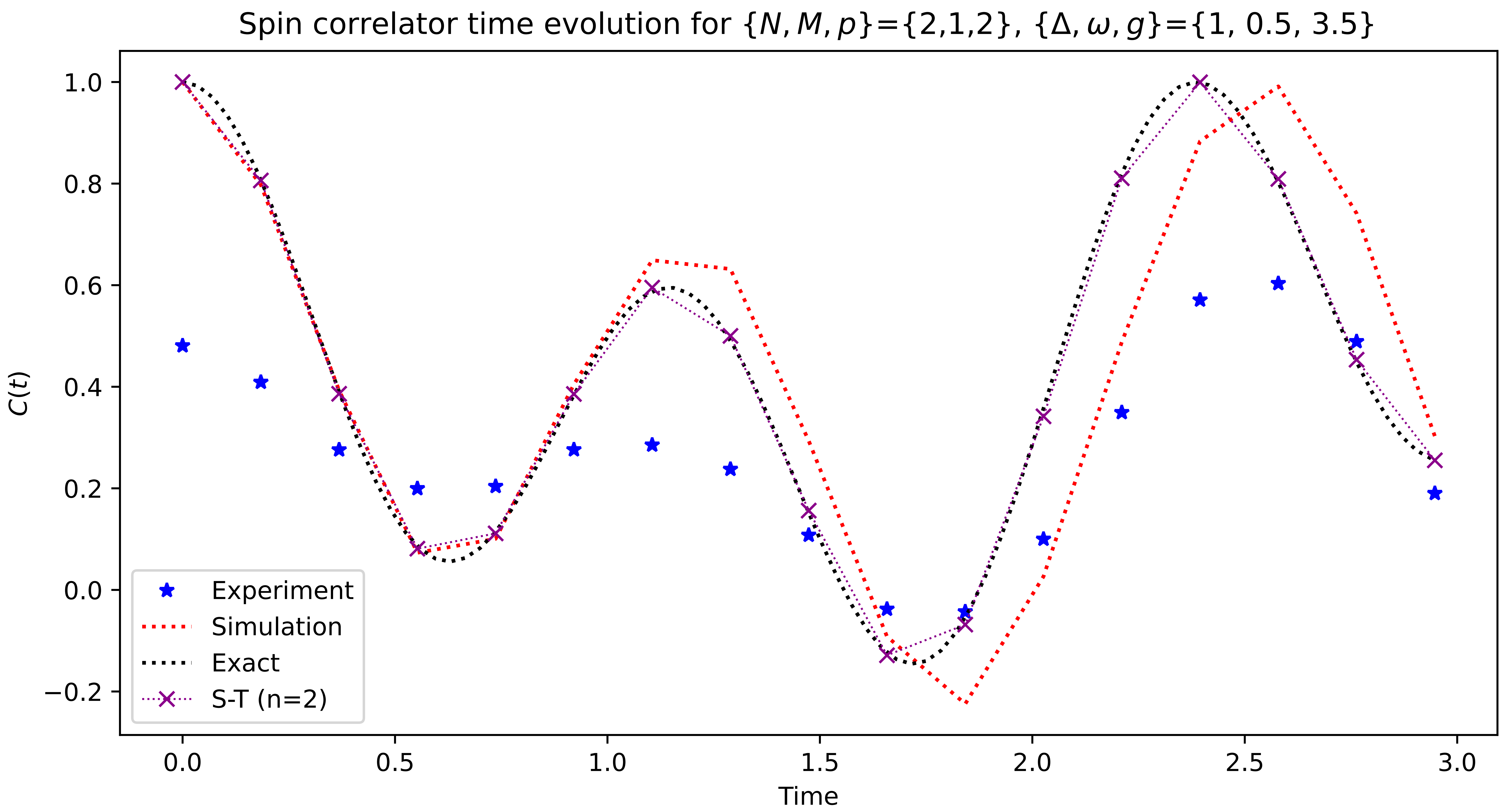
Results (Initial state |100> - excited boson)



Results (Initial state |100> - excited boson)

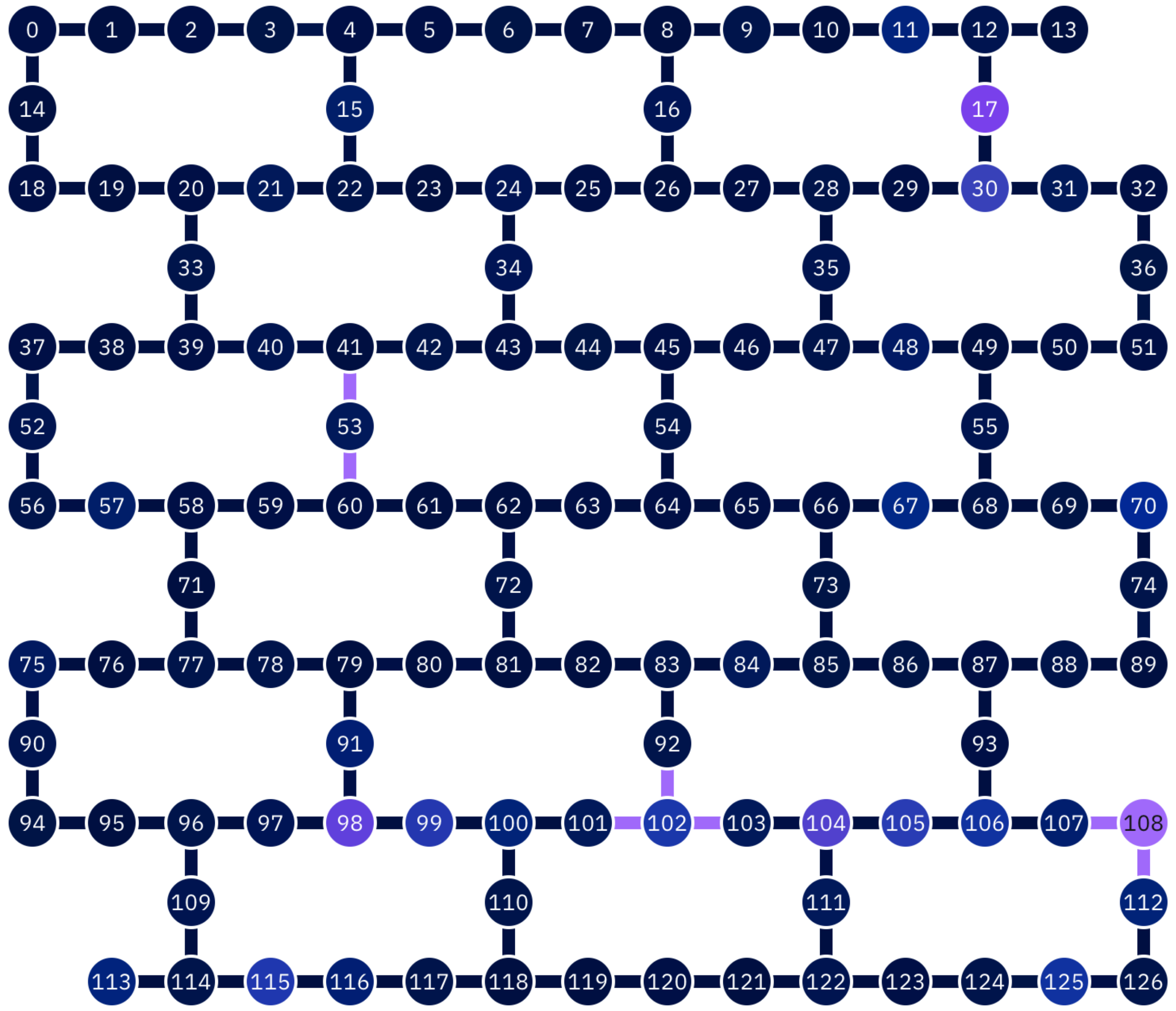


Results (Initial state |100> - excited boson)



Mapping

- We need to map to the hardware in an intelligent way to minimize SWAPs.



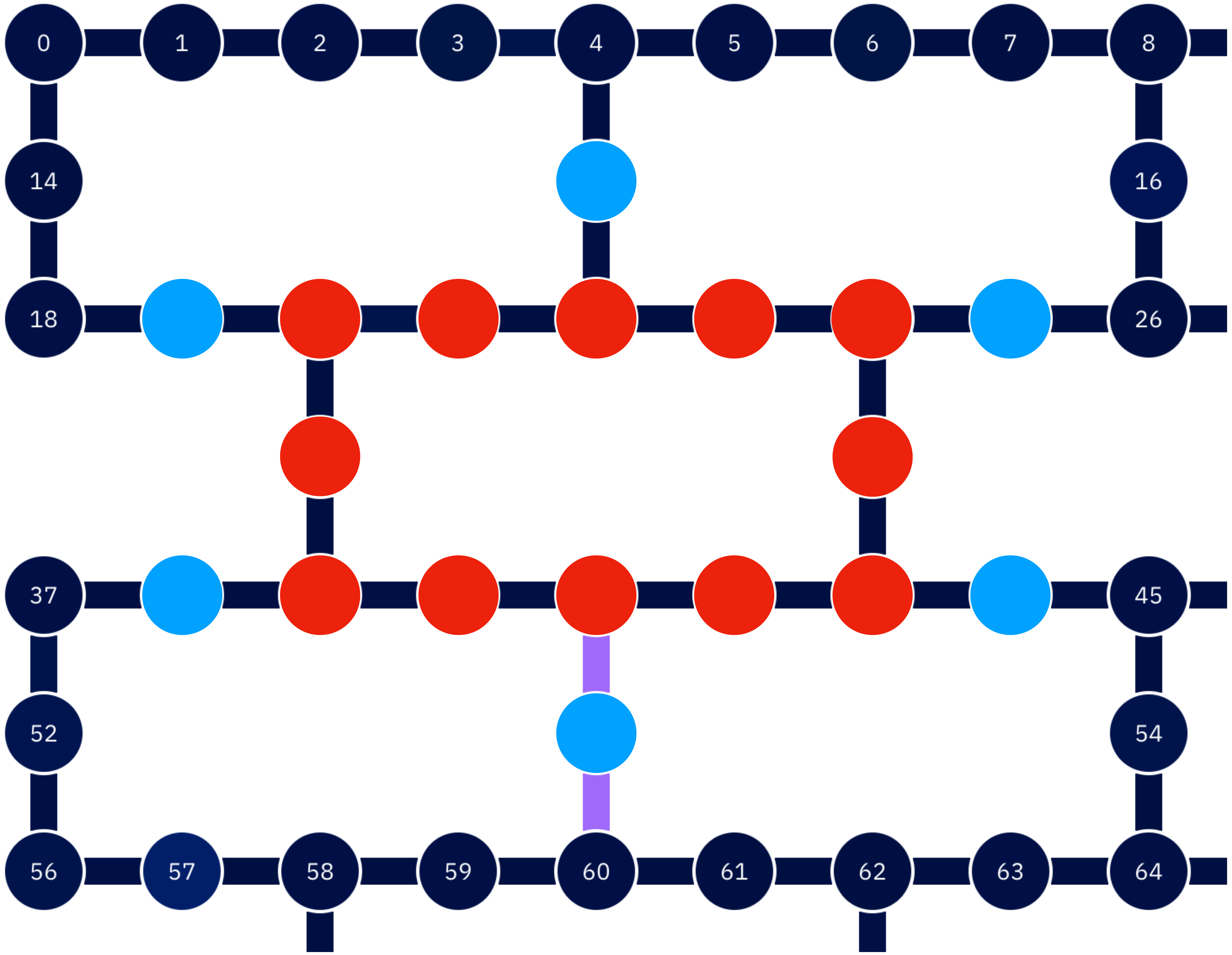
- For systems with larger occupation modes it is better to use unary encoding.

Mapping

- Unary encoding minimizes the number of SWAP gates and is more easily scalable.

$$N = 6$$

$$M = 1, p = 11$$



Thank You

Rensselaer

why not change the world?®

IBM