Creating a quantum computation simulation framework

Vladislav Guschakowski, Anantha Vasudevan

Friedrich-Schiller-Universität Jena, Max-Wien-Platz 1, 07743 Jena, Germany

GitHub[1]

FRIEDRICH-SCHILLER-UNIVERSITÄT JENA

Motivation

Powerful quantum algorithms, such as Shor's and Grover's, were already discovered over three decades ago. To this day the experimental realization of these algorithms still presents a large obstacle, due to high error rate in qubits. Thus, classical simulations of quantum circuits are still necessary[2].

Our goal is to create a **modular** and **simple-to-use Python** framework for **simulating quantum circuits**. The framework should be organized in a **robust** manner, with cooperative development and feature expansions in mind.

Our project focuses on two aspects:

1. **Learning how to create and maintain usable and expandable software frameworks in a research context.**
2. **Applying the framework to investigate the effect of noise on the Deutsch-Josza algorithm.**

Framework

Tools & Workflow

Collaborative coding projects require a **strong maintenance structure**. Helpful practices for this purpose are:

- **Version control**: logging changes between versions with the possibility to create development branches (**GitHub**)
- **Unit testing**: testing single units of a framework, thus assuring all former features don't break during updates
- **Jupyter Notebooks**: prepared coding environments showing how to use a tool and allowing for quick experimentation

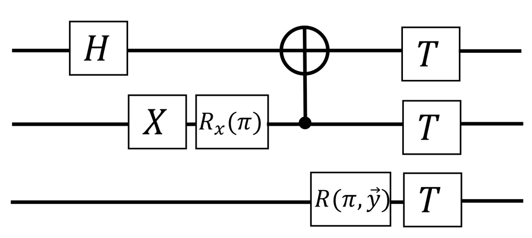
Features

- Fully works with **pure** & **mixed states** as **numpy arrays**
- Native **gate** & **channel application**
- Quick and flexible **instruction list syntax** using the internal ***instruction*** class

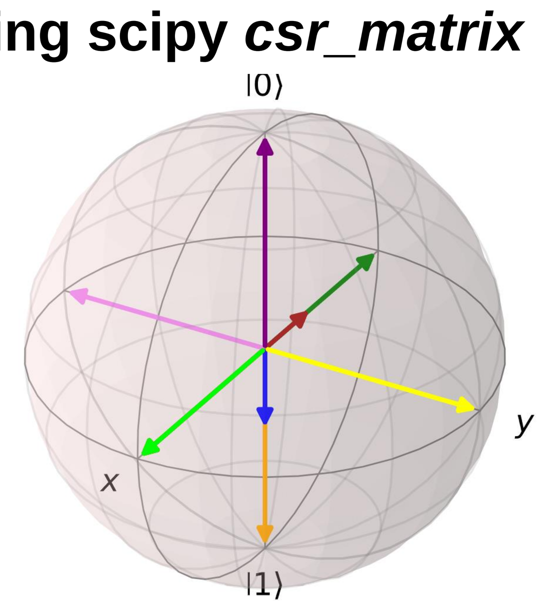
```
instructions = go.create_instruction_list(
                [["H",[1]],
                 ["X",[2]],
                 ["Rx",[2],np.pi],
                 ["CNOT",[2,1]],
                 ["R",[3],np.pi,np.array([0,1,0])],
                 ["T",[1,2,3]]])

state = reduce(go.apply_instruction, instructions, state)
```

applies



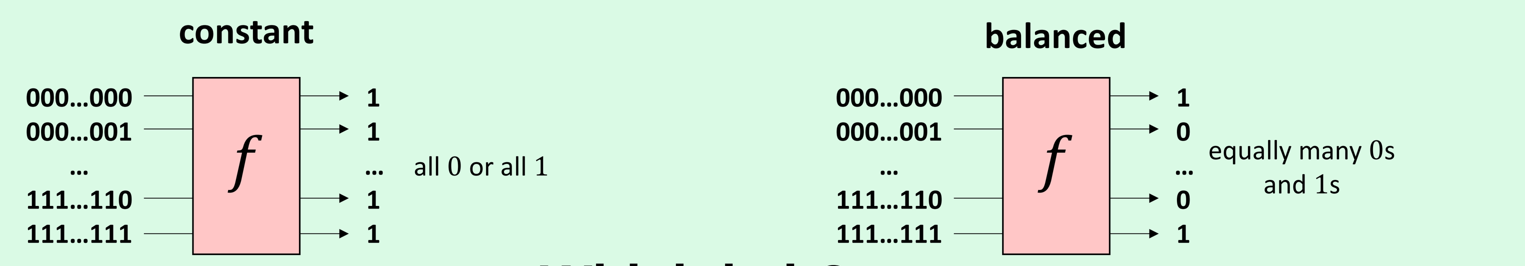
- Supports **sparse matrices** for pure states **using scipy *csr_matrix***
- Supports **projective measurements** for any given set of projectors
- Contains functions to create **Bloch plots** using **Qutip**
- Contains functions to initialize **Haar**, **Hilbert-Schmidt** and **Bures random states**[3]
- Thus, classical simulations of quantum circuits by **using sparse matrices**



Deutsch - Josza algorithm

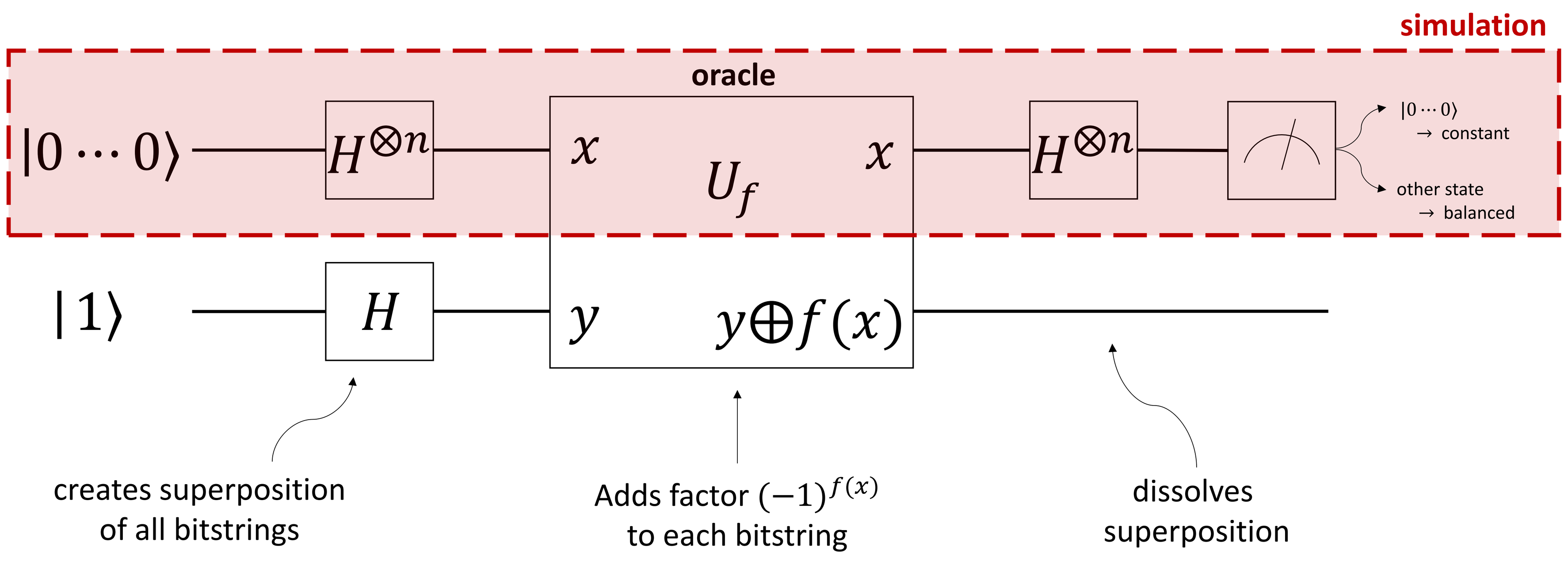
Problem

The function $f: \{0,1\}^n \to \{0,1\}$ takes bitstrings of length n and returns either 0 or 1. It is known to be either **constant** or **balanced**.



Which is it?

- **Classically** we need $\mathbf{2^{n-1}+1}$ **quarries** to so solve this problem, by direct testing
- The quantum algorithm known as **Deutsch-Josza algorithm** solves this problem in exactly **one quarry**

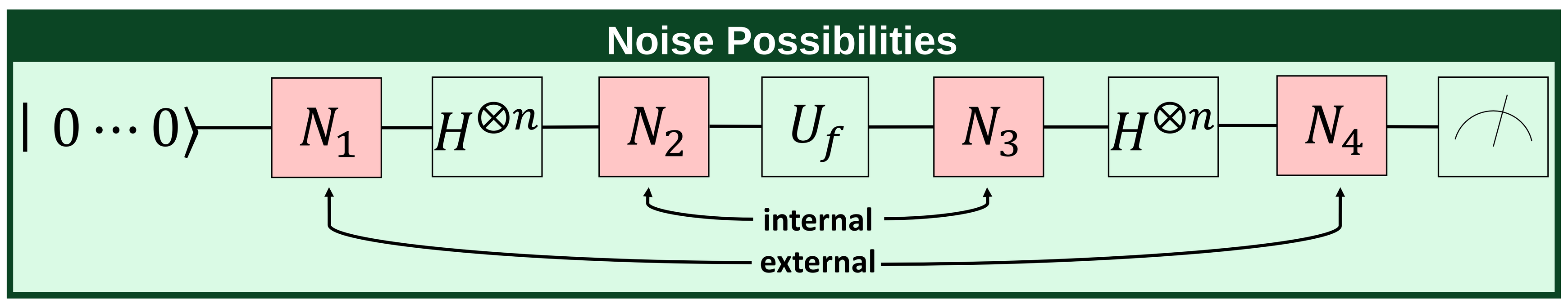


- If f is **constant** the measured state only received a factor of either 1 or -1 (which is just a phase) during this protocol, leading to the **final state being just $|0\cdots0\rangle$**
- If f is **balanced** the equally many bitstrings receive a factor of either 1 or -1, leading to **$|0\cdots0\rangle$ being canceled out before measuring**

Noisy Deutsch - Josza

To **investigate the effect of noise** on the Deutsch-Josza algorithm we used a black box function that simulated the oracle, while applying **single-qubit channels N_i at all the possible noise positions**.

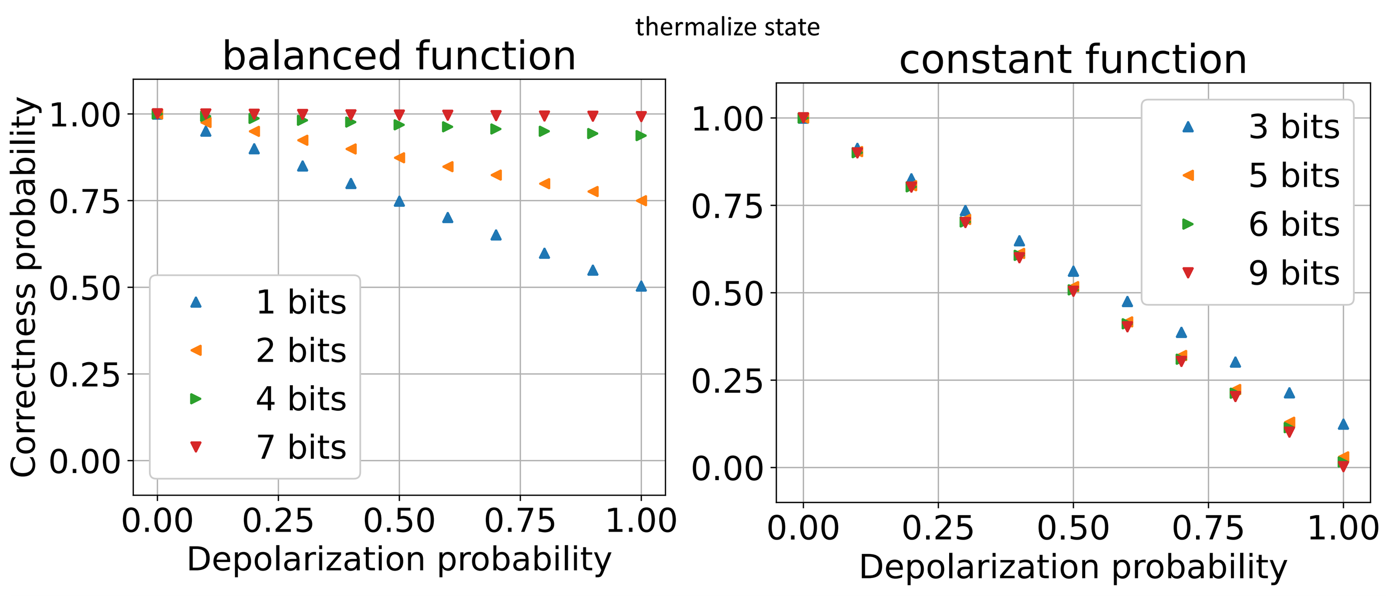
Noise Possibilities



Blackbox function

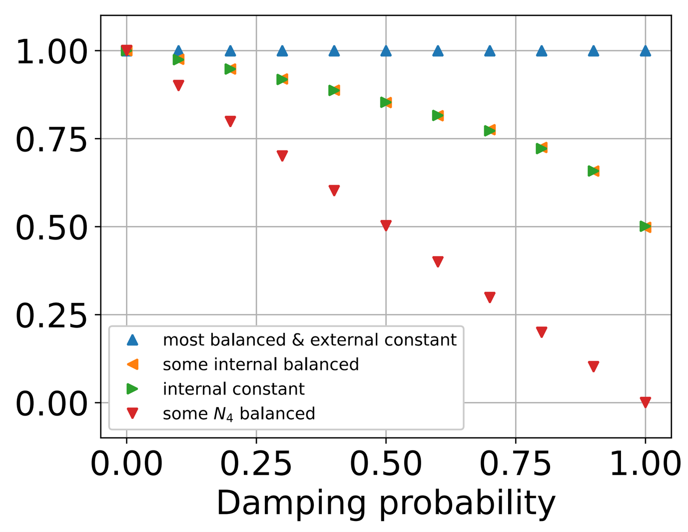
```python
def black_box(x):
    n=len(x)
    if n % 3 == 0:
        return 0
    if n % 5 == 0:
        return 1
    else:
        return int(x[n // 2])
```

Depolarization channel

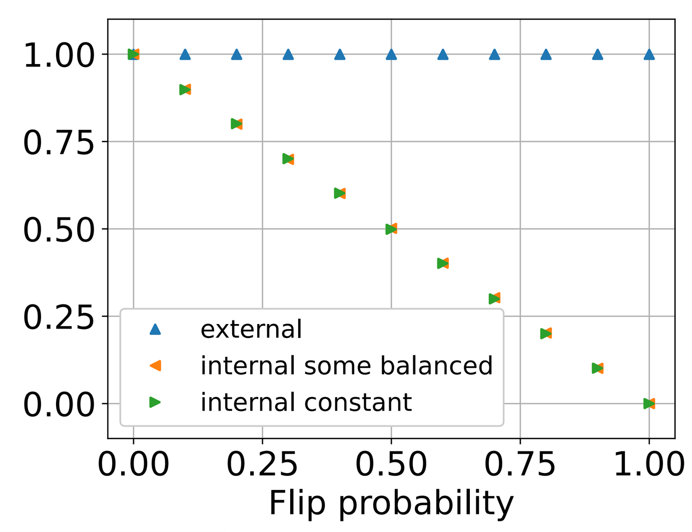


Amplitude damping channel

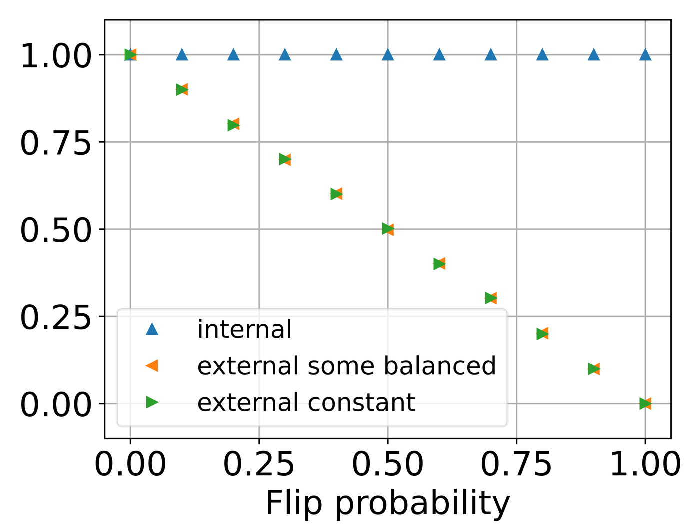
Pull qubit towards |0⟩



Bitflip channel



Phaseflip channel



Main results

It is more likely for the $|0\cdots0\rangle$ state to decohere than for it to be produced. Thus, we find:

- **Constant functions are more prone to errors**
- **Longer bitstrings stabilize balanced** functions to the **depolarization** channel
- **Longer bitstrings destabilize constant** functions to the **depolarization** channel

Internal noise positions act on equal superposition states, while external positions act on computational basis states, we therefore conclude:

- **Bitflips** are only relevant **externally**
- **Phaseflips** are only relevant **internally**

Why are not all balanced configurations affected?

For the balanced function to be **incorrectly estimated**, the output state needs to have some **overlap with the $|0\cdots0\rangle$ state**. Some noise channels do **not produce any overlap** with the $|0\cdots0\rangle$ state, for example, the bitflip channel on the first qubit of $|01\rangle$ will not impact the balanced function.

[1] **Guschakowski, Vladislav; Vasuvedan, Anantha. "quantum-computer-simulation." *GitHub Repository*.**
[2] Nielsen, Michael A., and Isaac L. Chuang. Quantum computation and quantum information. Cambridge University Press, 2010.
[3] Alhambra, Álvaro M. "Quantum many-body systems in thermal equilibrium." PRX Quantum 4.4 (2023): 040201.

vladislav.guschakowski@uni-jena.de
anantha.vasudevan@uni-jena.de