Evolving quantum circuits to implement stochastic and deterministic CA rules

S. Bhandari[1,2,3], S. Overskott[1,2,3], I. Adamopoulos[1,2,3], P. Lind[1,2,3], S. Senysov[1,3], S. Nichele[1,2,3,4,5]

[1]Oslo Metropolitan University, [2]AI Lab-Oslomet, [3]NordSTAR, [4]Simula Metropolitan Center for Digital Engineering, [5]Østfold University College

September 2022

Outline

- Introduction and Background
- The Genetic Algorithm to Evolve Quantum Circuits
- Implementation
- Simulation Results
- Summary

Quantum Computing

Quantum computing → a field in solving the limitations of classical computation

Quantum Computing

Quantum computing → a field in solving the limitations of classical computation

Components of quantum computation

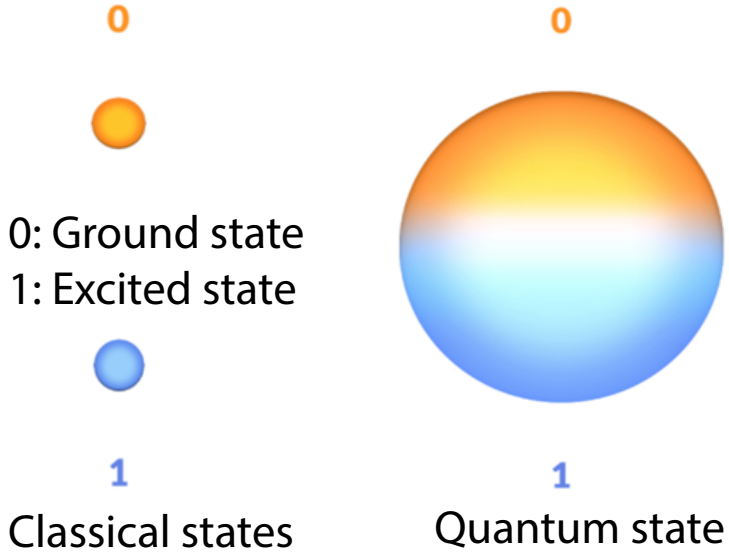
- Qubits
- Quantum gates
- Quantum circuits

Qubits

- Unlike a normal computer bit, which can be 0 or 1, qubits can have an infinite number of possible states if their state is not measured.

Qubits

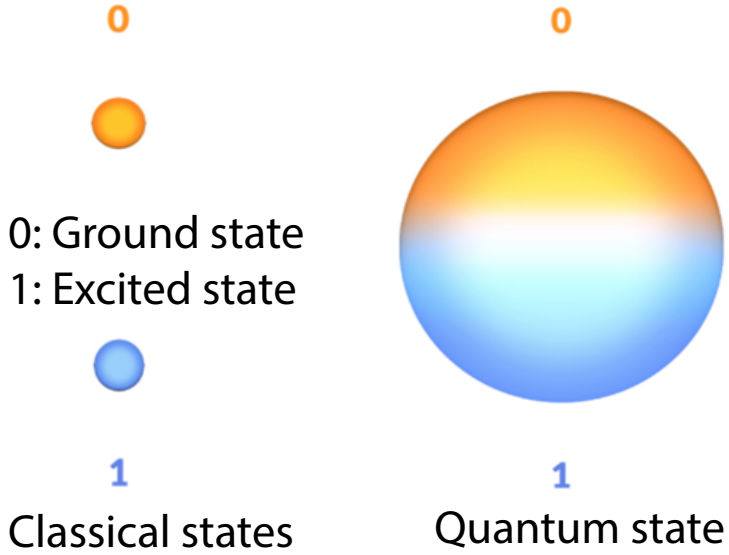
- Unlike a normal computer bit, which can be 0 or 1, qubits can have an infinite number of possible states if their state is not measured.



The differences between the classical and the quantum states. [1]

Qubits

- Unlike a normal computer bit, which can be 0 or 1, qubits can have an infinite number of possible states if their state is not measured.



The differences between the classical and the quantum states. [1]

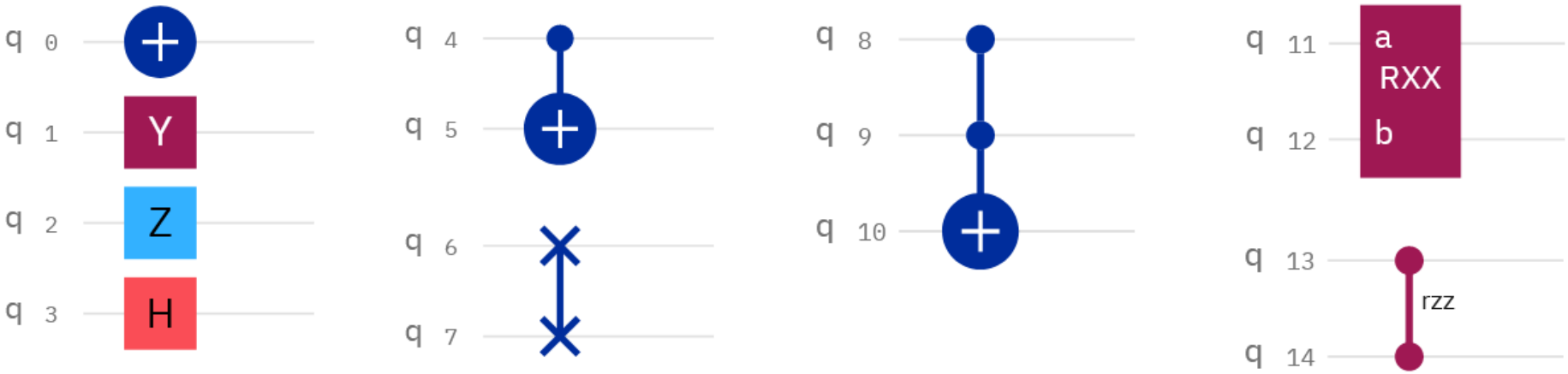
- Question???
- What happened if all the possible number of states of qubits are measured?

Quantum Gates→ operates on quantum bits

- Two key aspect that differentiate classical gates: *superposition and entanglement*
- Quantum gates are reversible

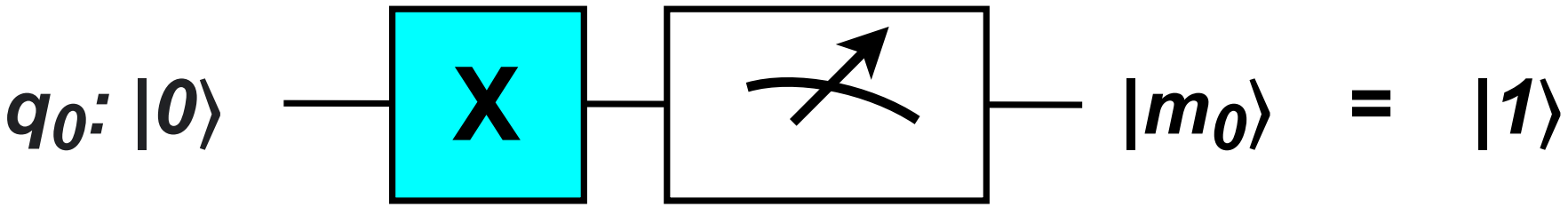
Quantum Gates→ operates on quantum bits

- Two key aspect that differentiate classical gates: *superposition and entanglement*
- Quantum gates are reversible



An overview of quantum gates available to the algorithm: X-gate (q0), Y-gate (q1), Z-gate (q2), Hadamard gate (q3), CNOT (control: q4, target: q5), swap-gate (q6, q7), Toffoli-gate (control: q8 and q9; target:q10), RXX-gate (q11, q12), RZZ-gate (q13, q14) [2].

Quantum Circuits→ collection of quantum gates, measurements and resets, all of which may be conditioned on and use data from the real-time classical computation.



Example of a single qubit and depth one quantum circuit

Cellular Automata

→discrete dynamical system that consists of an infinite array of cells.

→each cell has a state from a finite state set

Cellular Automata

→discrete dynamical system that consists of an infinite array of cells.

→each cell has a state from a finite state set

Elementary CA → state cell {0,1} which are updated according to one out of 256 possible rules matching each one of the eight neighborhoods

Cellular Automata

→discrete dynamical system that consists of an infinite array of cells.

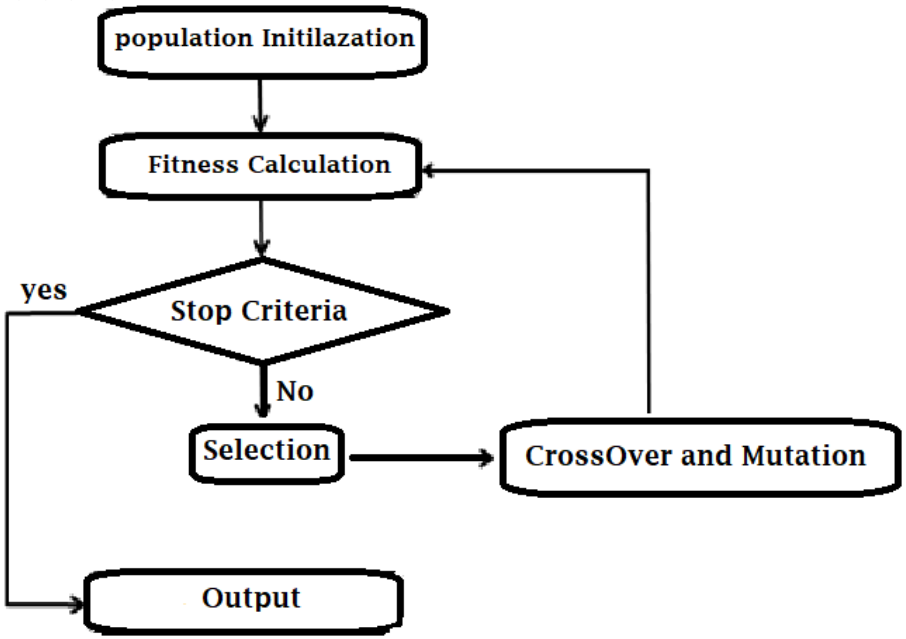
→each cell has a state from a finite state set

Elementary CA → state cell {0,1} which are updated according to one out of 256 possible rules matching each one of the eight neighborhoods

Neighbors	Stoc. CAProb[3]	Rule90	Rule110	Ran. Prob. 1	Ran. Prob. 2	Ran. Prob. 3
[0, 0, 0]	0.394221	0	0	0.6364	0.4778	0.1988
[0, 0, 1]	0.094721	1	1	0.6603	0.5604	0.4701
[0, 1, 0]	0.239492	0	1	0.5261	0.8528	0.9836
[0, 1, 1]	0.408455	1	1	0.1748	0.4818	0.7115
[1, 0, 0]	0	1	0	0.8820	0.3143	0.6616
[1, 0, 1]	0.730203	0	1	0.3371	0.3464	0.1218
[1, 1, 0]	0.915034	1	1	0.0340	0.0678	0.1328
[1, 1, 1]	1	0	0	0.4444	0.9124	0.7306

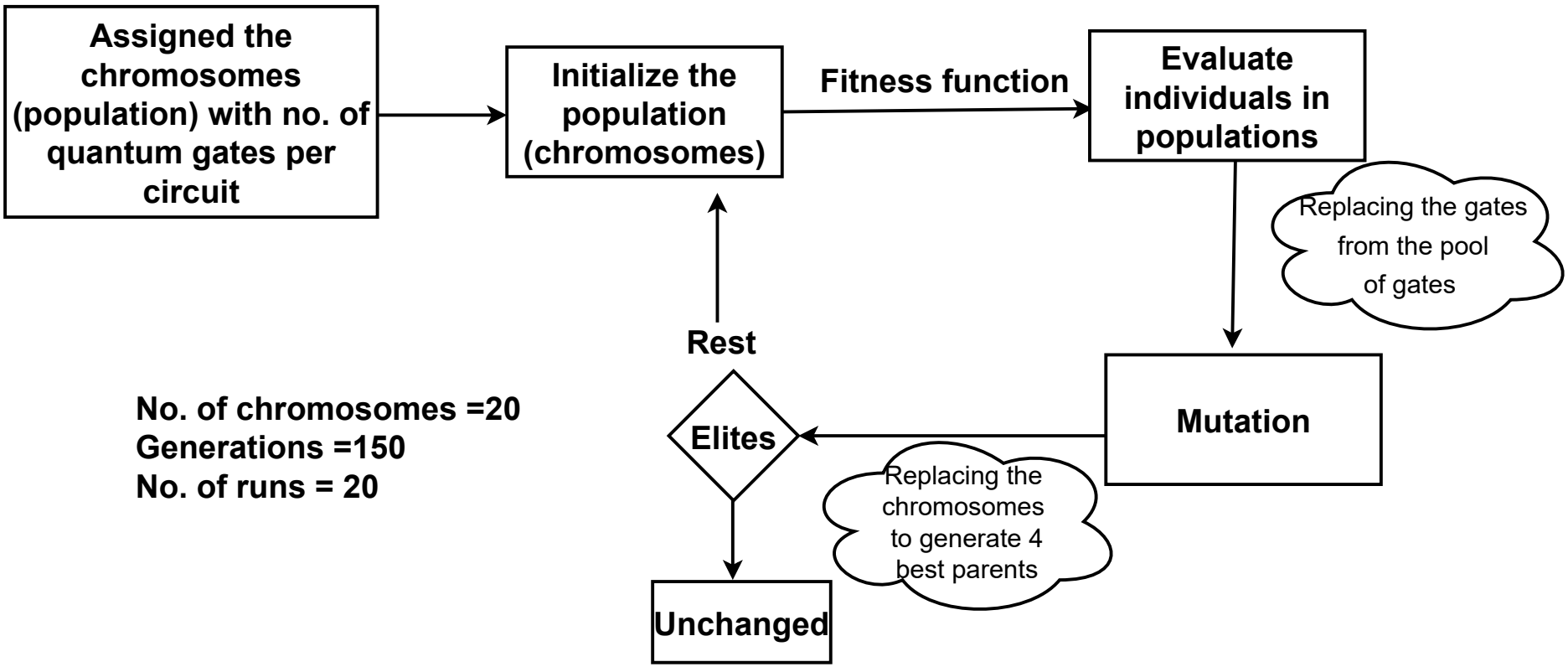
Genetic Algorithm (GA)

- GA $\rightarrow$ a search heuristic inspired by Charles Darwin theory of natural selection.



Decision tree for genetic algorithm

Genetic Algorithm to Evolve QC



Decision tree for genetic algorithm to evolve quantum circuits



Fitness Function

- the sum of absolute difference between each $P(\omega)$ and corresponding $Q(\omega)$

$$F = \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)| \,. \tag{1}$$

Fitness Function

- the sum of absolute difference between each $P(\omega)$ and corresponding $Q(\omega)$

$$F = \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)| \,. \tag{1}$$

- Kullback-Leibler (KL) divergence

$$D_{KL}(P||Q) = \sum_{\omega \in \Omega} P(\omega) log\Big(\frac{P(\omega)}{Q(\omega)}\Big) \,. \tag{2}$$



Fitness Function

- the sum of absolute difference between each $P(\omega)$ and corresponding $Q(\omega)$

$$F = \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)| \,. \tag{1}$$

- Kullback-Leibler (KL) divergence

$$D_{KL}(P||Q) = \sum_{\omega \in \Omega} P(\omega) log\Big(\frac{P(\omega)}{Q(\omega)}\Big) \,. \tag{2}$$

Difference→the former treats each probability as pure values → the latter considers deviations of higher probability with a stronger weight than low probability values

Implementation

The project was created and done in python version 3.8 with Qiskit version 0.34.1 and SciPy.special 1.7.1.

- Qiskit → an open source quantum development tool for working with quantum computers at the level of pulses, circuits, and application modules.

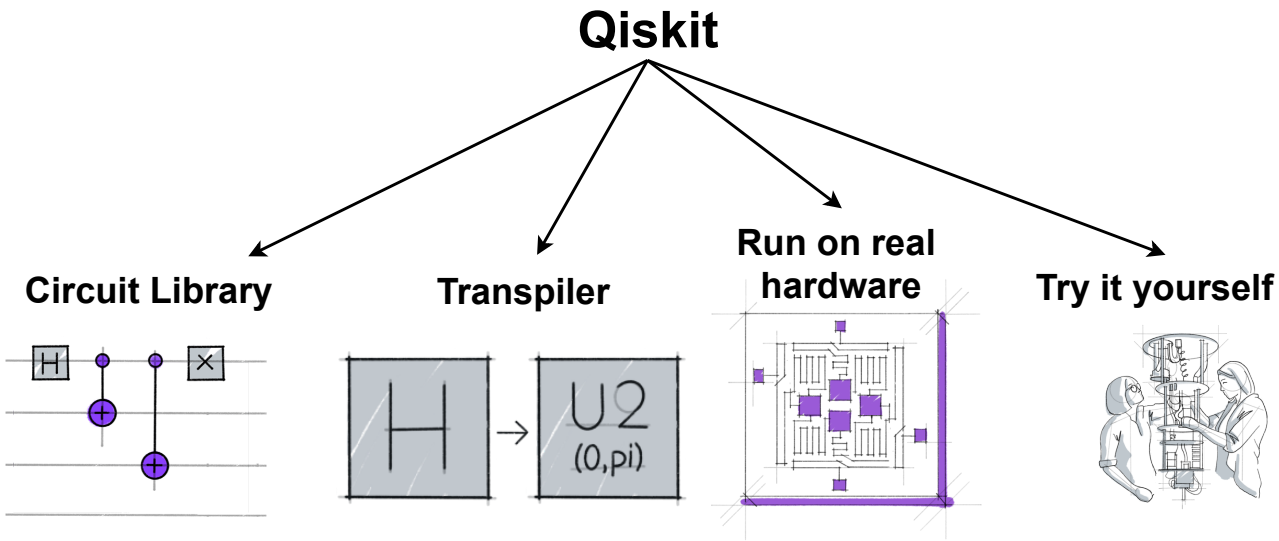


Figure source: qiskit.org

Implementation

The EA module consist of three classes

- **Chromosome**: all functions for generating and mutating the lists.

Implementation

The EA module consist of three classes

- **Chromosome**: all functions for generating and mutating the lists.
- **Generation**: Stores several chromosomes. It can populate itself with a given number of generated chromosomes. Function to compare the fitness and store the fitness list.

Implementation

The EA module consist of three classes

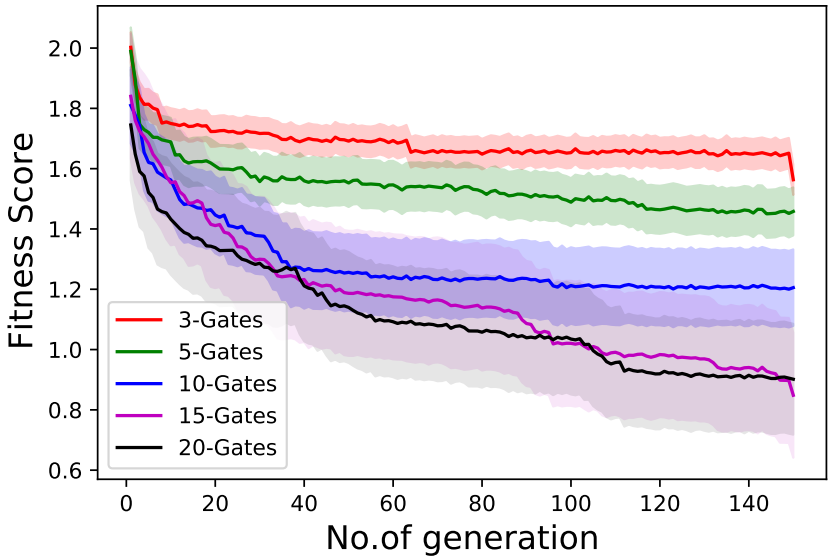
- **Chromosome**: all functions for generating and mutating the lists.
- **Generation**: Stores several chromosomes. It can populate itself with a given number of generated chromosomes. Function to compare the fitness and store the fitness list.
- **Circuits**: Handles the parsing chromosome list to circuit and initializing Qiskit code to simulate the circuits.

Github Link: https://github.com/shailendrabhandari/QUEVO

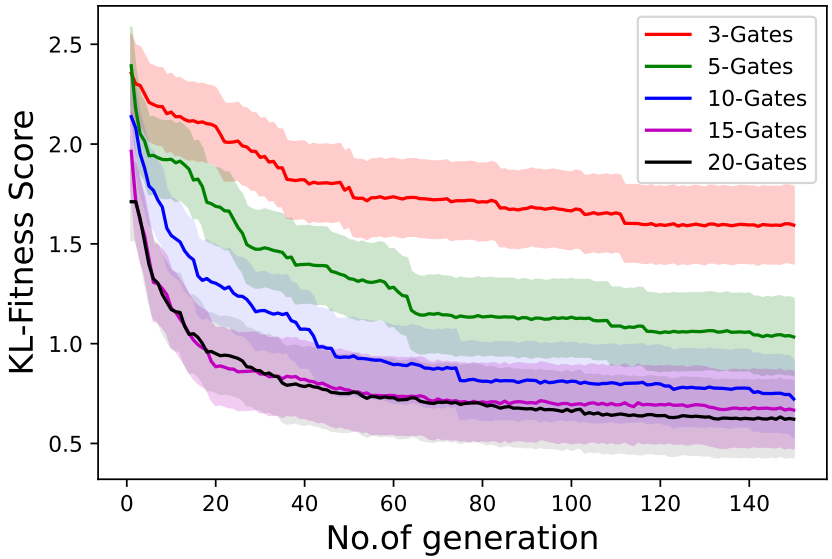
Experimental results

Stochastic Critical Cellular Automata

→ the mutation is fixed to 10% and considered the number of gates used

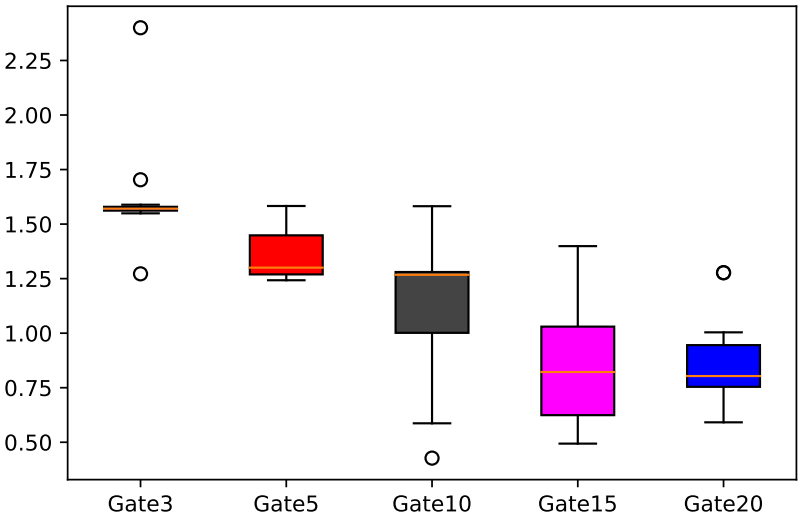


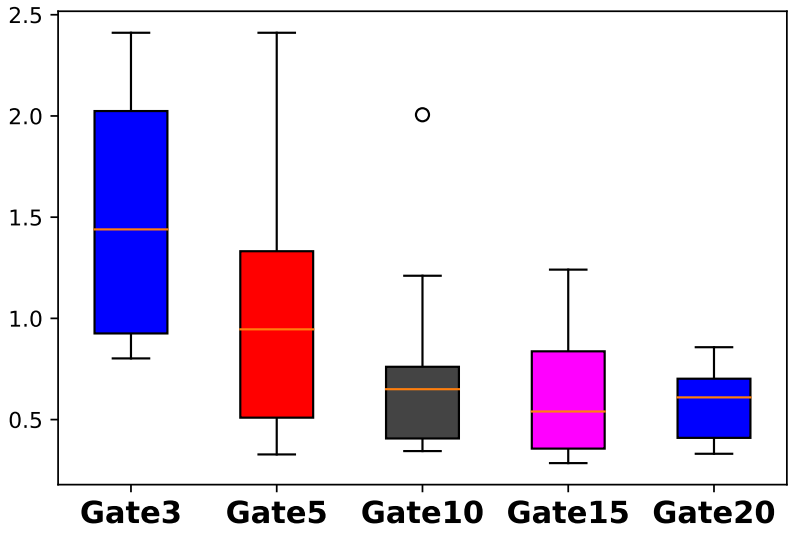
For the absolute sum of differences, Eq. (1)



Kullback-Leibler divergence, Eq. (2)

Experimental results





For the absolute sum of differences, Eq. (1)

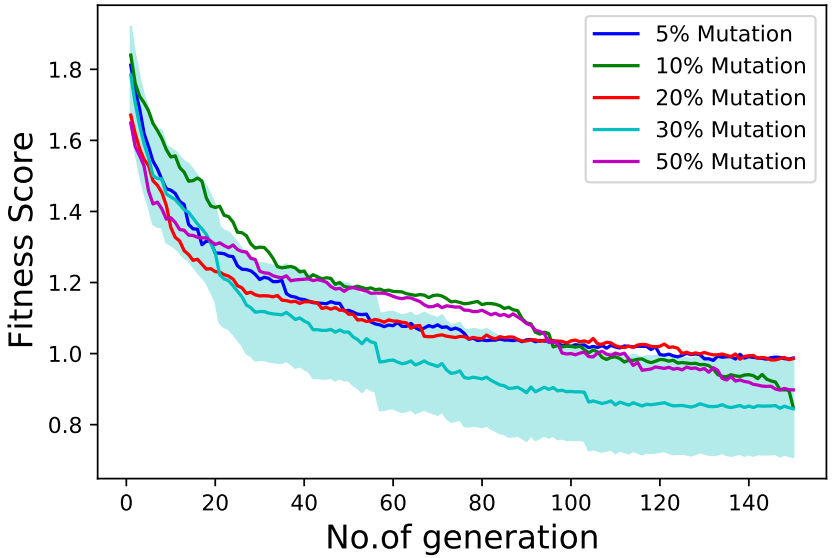
Kullback-Leibler divergence, Eq. (2)

The fitness scores of each gates for the box plots are the best fitness scores per run.

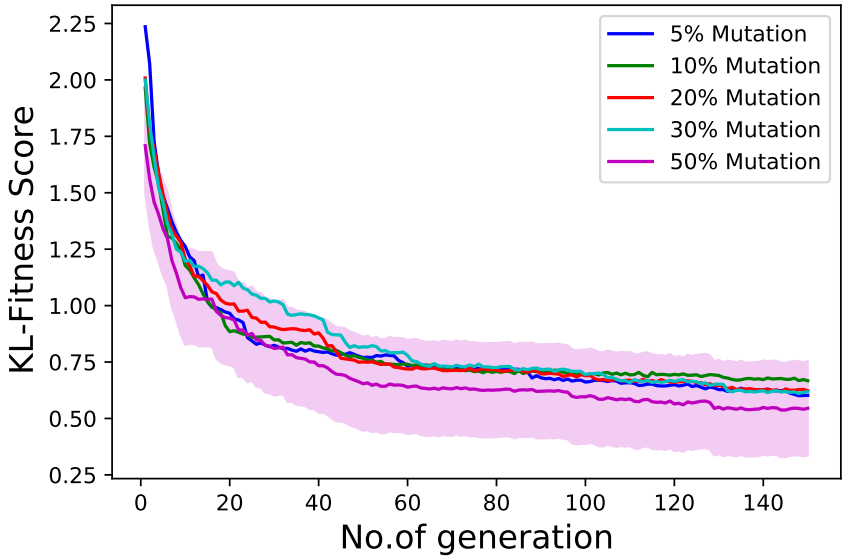
Experimental results

Stochastic Critical Cellular Automata

$\rightarrow$ the number of gates is 15 and considered the mutation from 5-50%



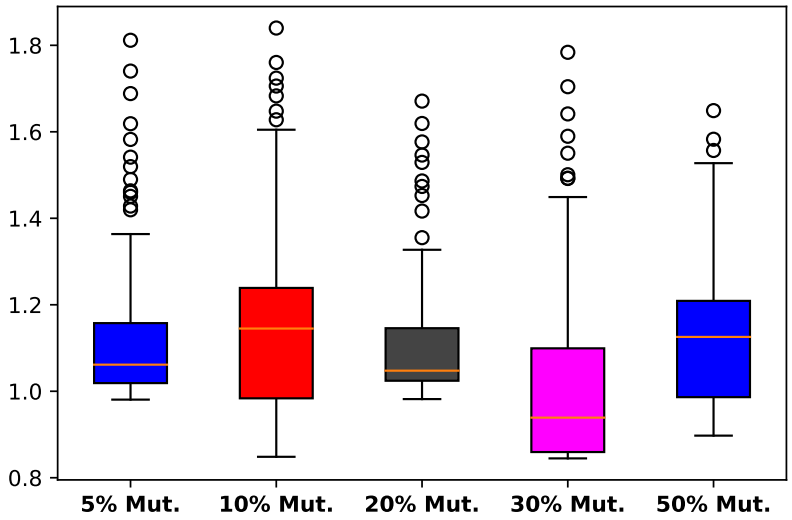
For the absolute sum of differences, Eq. (1)

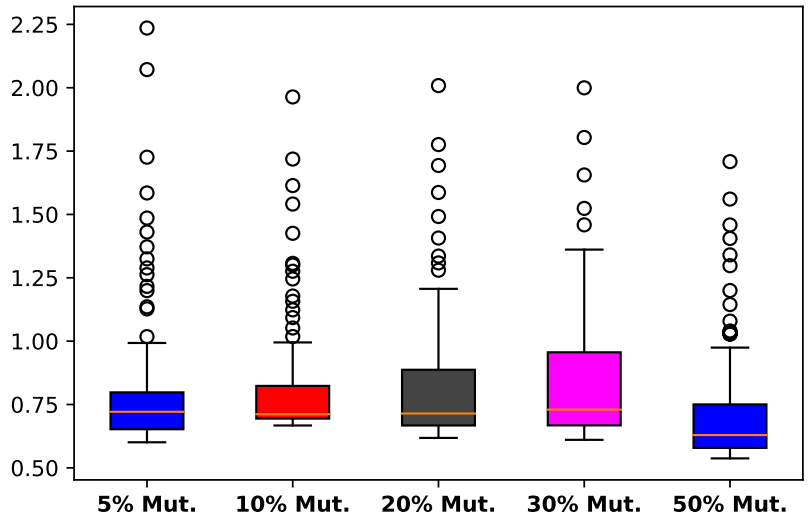


Kullback-Leibler divergence, Eq. (2)

The fitness scores as a function of the number of generations, for different mutation rates.

Experimental results





For the absolute sum of differences, Eq. (1)

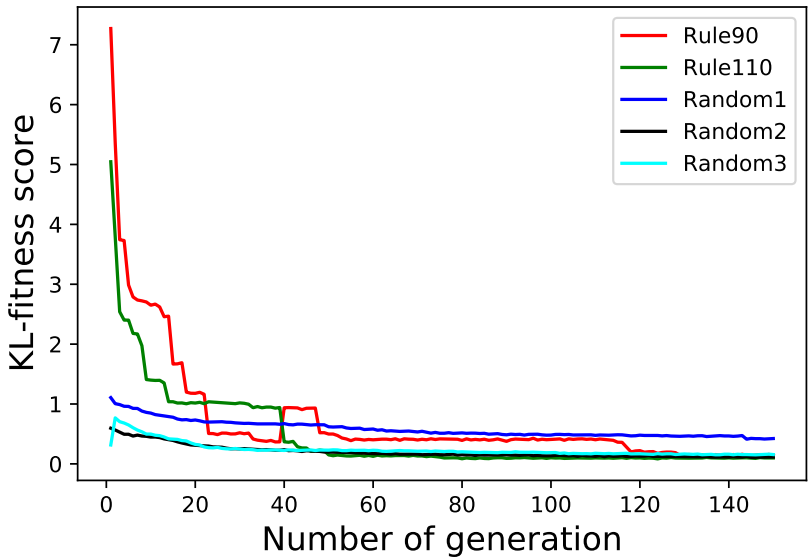
Kullback-Leibler divergence, Eq. (2)

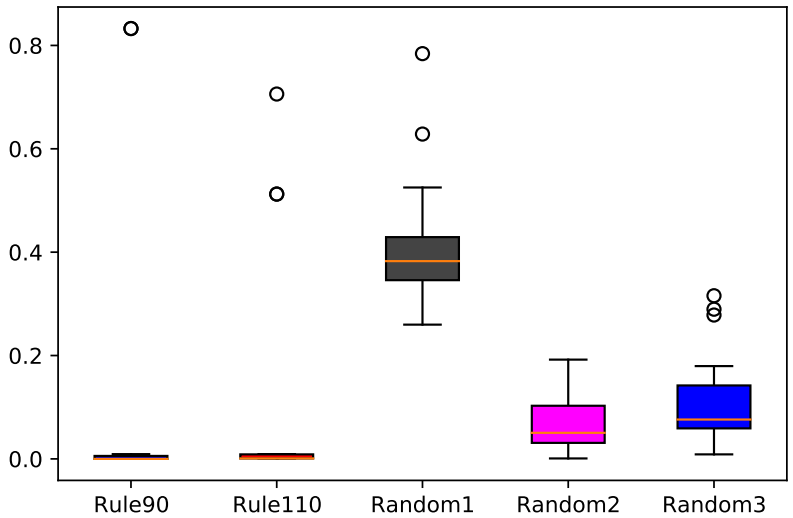
Mutation rates vs. the best fitness scores. The fitness score is the best fitness scores for 20 runs

Experimental results

The deterministic and the Stochastic CA rules

$\rightarrow$ 10% mutation probability, 15 number of gates and the KL-fitness function

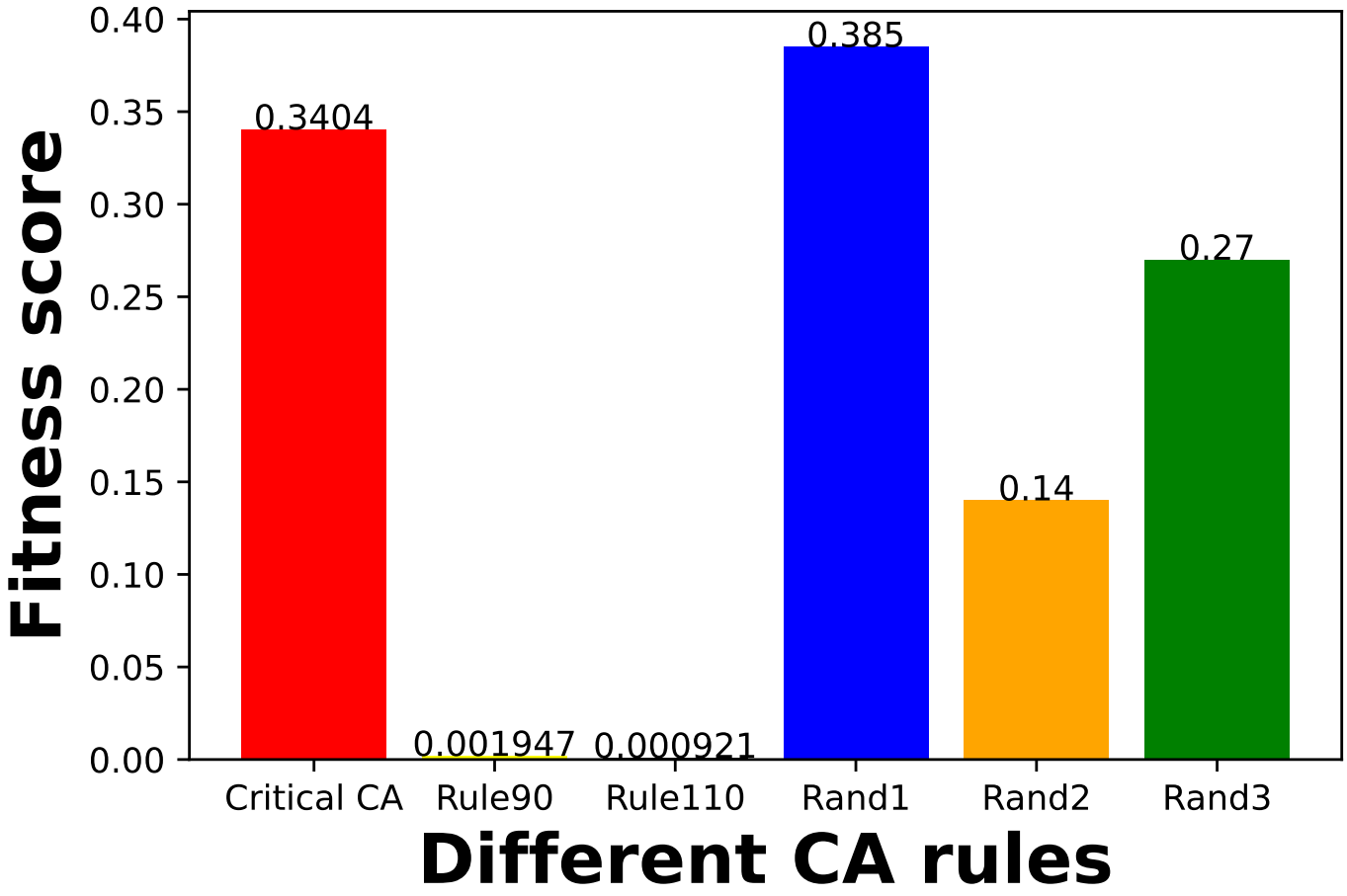




Fitness scores for different sets of probabilities against the number of generations

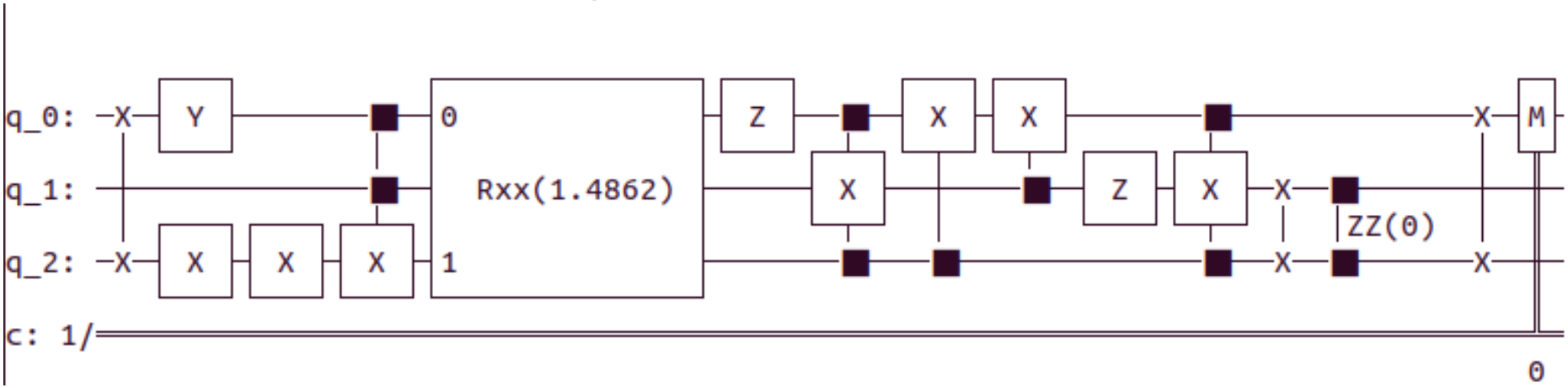
Experimental results

Comparison of the the best fitness scores for all three CA rules



Experimental results

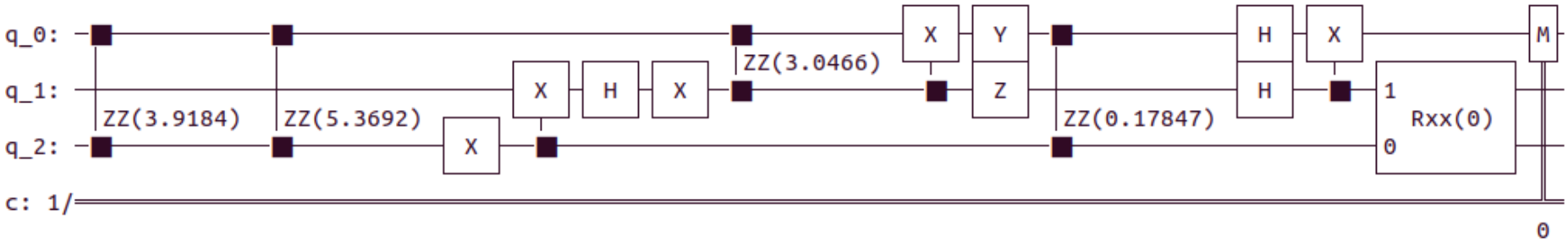
For critical stochastic CA



Visualization of a circuit created by 15 number of gates, 10% mutation probability with KL fitness scores.

Experimental results

For deterministic CA (Rule 110)



Visualization of a circuit created by 15 number of gates, 10% mutation probability with KL fitness scores

Summary

- A simple framework to derive quantum circuits reproducing specific CA rules, using genetic algorithms has been studied.

Summary

- A simple framework to derive quantum circuits reproducing specific CA rules, using genetic algorithms has been studied.
- The differences observed when changing the number of gates and the mutation rate has been inspected.

Summary

- A simple framework to derive quantum circuits reproducing specific CA rules, using genetic algorithms has been studied.
- The differences observed when changing the number of gates and the mutation rate has been inspected.
- We benchmark our methods with stochastic as well as deterministic CA rules.

Future Work

- More experiments on stochastic cellular automata
- Additional genetic operator: Crossover

References

[1] F. Jazaeri., A. Beckers., A. Tajalli., J. Sallese. (2019). A Review on Quantum Computing: Qubits, Cryogenic Electronics and Cryogenic MOSFET Physics, arXiv, eprint: 1908.02656.

[2] Operations glossary.https://quantum-computing.ibm.com/composer/docs/iqx/operations_ glossary.

[3] Pontes-Filho, S., et al.: A neuro-inspired general framework for the evolution of stochastic dynamical systems: cellular automata, random Boolean networks and echo state networks towards criticality. Cogn. Neurodyn. 14(5), 657–674 (2020). https://doi.org/10.1007/s11571-020-09600-x

Thank you for your attention

