Comparing Classical and Quantum Finite Automata

Kaitlin Gili,[1,2] Rudy Raymond[3], Rodney Van Meter[3], and Kohei M. Itoh[3]

[1] *Dept. of Physics, Stevens Institute of Technology, Hoboken, NJ, USA*
[2]*Nakatani RIES: Research & International Experiences for Students Fellowship in Japan, Nakatani Foundation, Tokyo, Japan*
[3]*IBM Q Hub, Keio University, Tokyo, Japan*

Quantum computers offer the advantage of running more efficient algorithms because they are composed of qubits rather than conventional bits that are used in classical computers. This motivates the exploration of quantum algorithms and models with the expectation that they can solve many more complex problems. Here, we are exploring one of the simplest models of computation by comparing Deterministic Finite Automata and Quantum Finite Automata (Ambainis and Freivalds, 1998). This research focuses on applying quantum principles to the following problem: Consider a string a^i with i letters. We want to determine whether the string is in the language L where $L = \{a^i \mid i \text{ is divisible by } p\}$ and p is a given prime number. If i is divisible by p, we accept the string into the language, and if not, we reject it. $|0\rangle$ and $|1\rangle$ qubit states serve as the accept and reject states. Classically, using the highest known prime integer, this algorithm requires a minimum of 77,232,917 bits, whereas the quantum finite automata only requires 27 qubits. Using Python's Quantum Information Software Kit (Qiskit), I have implemented a program [1] that can determine if the length of a string is divisible by a large prime number with exponentially fewer qubits, thus further demonstrating the potential of quantum models.

[1] https://github.com/kaitlinmgili/sharing-github/blob/master/Quantum%20Finite%20Automata%20Algorithm%20-2.ipynb

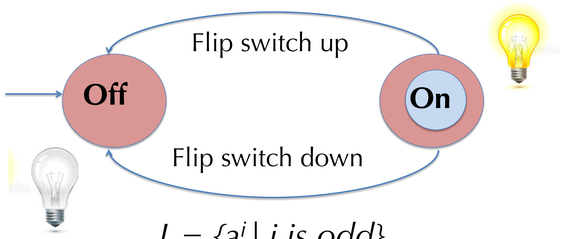
Comparing Classical and Quantum Finite Automata (QFA)

Kaitlin Gili,[1,2,3,4] Rudy Raymond,[3] Rodney Van Meter,[3] and Kohei M. Itoh[3,4]

[1]*Department of Physics, Stevens Institute of Technology,* [2]*Nakatani-RIES: Research and International Experiences for Students Fellowship, Rice University,* [3]*IBM Q Hub Research Center, Keio University,* [4]*Department of Applied Physics and Physico-Informatics Itoh Lab Group, Keio University*

Contact: kgili@stevens.edu

Classical Finite Automata



$L = \{a^i \mid i \text{ is odd}\}$

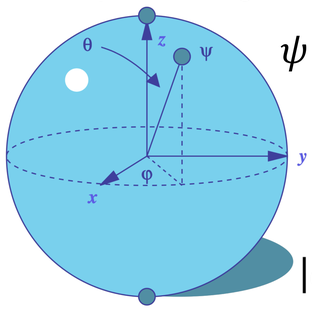
Goal: To accept or reject an input from some character set based on the pattern defined by the machine

Applications

- Traffic Lights
- Vending Machines
- Elevators
- Turnstiles

Quantum Finite Automata

Bloch Sphere ~ Single Qubit Representation



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

$$\alpha = cos\theta$$

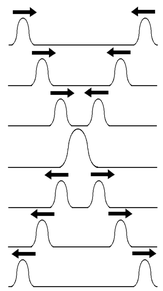
$$\beta = e^{i\varphi} sin\theta$$

$$|\alpha|^2 + |\beta|^2 = 1$$

Goal: To accept or reject an input from some character set using quantum properties such as superposition and the probability that the qubit is in a specific state

Motivation

- Interactive Proof Systems
- Demonstrate a quantum advantage in simple models
- Invention of quantum state machines (traffic lights, elevators, etc.)



Wave Superposition

The Problem: Prime Divisibility

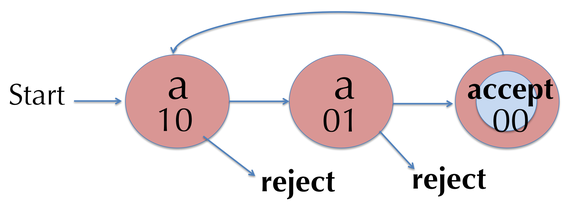
Consider a string with a^i letters and we want to know if the string is accepted into the language L, where $L = \{a^i \mid i \text{ is divisible by } p\}$ and p is a prime number[1]

Will qubits offer an advantage to this finite automata problem?

The Solution: Classical vs. Quantum

Example: String = "aaa", Prime = 3

Classical

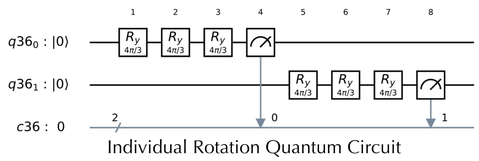


- Requires 2 bits & 3 states
- Runs an **individual** finite automata

Quantum

How can we use probability?

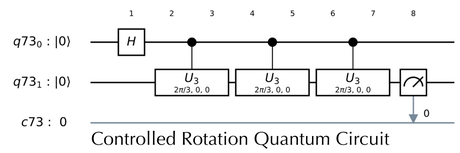
- $\alpha^2 = 1$ when $\theta = 2\pi ki/p$ and i divides p
- We **accept** when $\alpha^2 = 1$
- Otherwise, we **reject** with a probability of β^2
- Use Y-Rotation on individual qubits



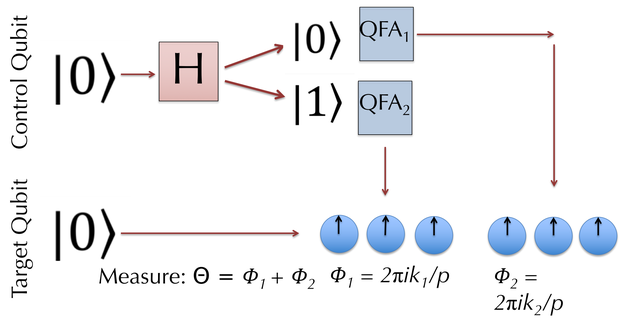
Individual Rotation Quantum Circuit

How can we use superposition?

- Use a Hadamard gate to create a superposition on control qubits
- Perform a Controlled Y-Rotation on target qubit



Controlled Rotation Quantum Circuit



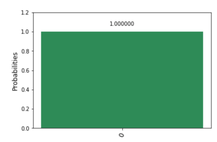
- Demonstrates that we can run **multiple** QFAs at the same time
- Therefore, we can use exponentially fewer qubits

QISKit Implementation in Python

QISKit python

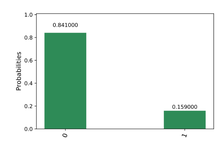
```
#Function that takes in a prime number and a string of letters and returns a quantum circuit
def qfa_controlled_algorithm(string, prime):
    if isprime(prime) == False:
        raise ValueError("This number is not a prime") #Raises a ValueError if the input prime number is not prime
    else:
        n = math.ceil((math.log(math.log(prime,2),2))) #Represents log(log(p)) control qubits
        states = 2 ** (n) #Number of states that the qubits can represent/Number of QFA's to be performed
        qr = QuantumRegister(n+1) #Creates a quantum register of log(log(prime)) control qubits + 1 target qubit
        cr = ClassicalRegister(1) #Creates a classical register of log(log(prime)) control qubits + 1 target qubit
        circuitName = "QuantumFiniteAutomata" #Name of the circuit/algorithm
        control_qfaCircuit = QuantumCircuit(qr, cr) #Defining the circuit to take in the values of qr and cr
        for q in range(n): #We want to take each control qubit and put them in a superposition by applying a Hadamard Ga
            control_qfaCircuit.h(qr[q])
        for letter in string: #For each letter in the string, we want to apply a series of Controlled Y-rotations
            for q in range(n):
                control_qfaCircuit.cu3(2*math.pi*(2**q)/prime, 0, 0, qr[q], qr[n]) #Controlled Y on Target qubit
        control_qfaCircuit.measure(qr[n], cr[0]) #Measure the target qubit
        return control_qfaCircuit #Returns the created quantum circuit
```

Local Simulator



- Probability of 1 of collapsing to zero
- Accepts "aaa" into L

Real Device



- Rejects "aaa" with an error of 16%
- Too much noise from 5-qubit system

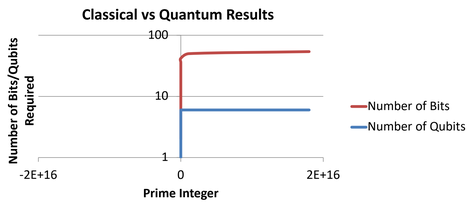
Conclusion

Classical

- Requires $log(p)$ bits and p states[1]
- Dividing by the largest known prime integer requires a minimum of **77,232,917 bits** to solve the problem

Quantum

- Requires $log(log(p))$ qubits and $log(p)$ states[1]
- Dividing by the largest known prime integer requires a minimum of **27 qubits** to solve the problem
- Requires **exponentially** less memory



Future Work

- Test for noise and use error correction techniques to improve the solution on the 5-qubit computer
- Test the solution with larger qubit systems such as IBM Q's 20-qubit computer
- Apply to real world finite automata

References

[1] Ambainis, A., & Freivalds, R. (1998). 1-way quantum finite automata: Strengths, weaknesses and generalizations. *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*. doi:10.1109/sfcs.1998.743469

Acknowledgements

This research project was conducted as part of the 2017 Nakatani RIES Fellowship for U.S. Students with funding from the Nakatani Foundation. For more information see: http://nakatani-ries.rice.edu/.
Special thanks to Prof. Kono, Sarah Phillips, Ogawa-san, Rudy Raymond, Rodney Van Meter, and Prof. Itoh.