Investigating Grover's Algorithm

Sydney Allen, Weston Benner, Liyan Feng, Andrew Liu, Andrei Perepelitsa, Jordan Tidwell, Sacha Uparipab

Mentor: Skandaprasad Rao

Introduction:

- Searching through unsorted data on a classical computer is slow
  - Complexity is $O(N)$, meaning every item must be checked once
- Quantum computers can search through data faster
  - **Grover's Algorithm** is considered the most efficient solution
  - Complexity is $O(\sqrt{N})$, which gives a significant speed-up

Objective:

- **Describe** the properties of several quantum gates
- **Pseudocode** Grover's algorithm
- **Simulate** Grover's Algorithm on a classical computer
- **Explore** practical applications of Grover's Algorithm

Background:

- **Qubits** store distributions of values **between 0 and 1**
  - This gives an advantage over bits, which store only **0's & 1's**
- This property allows for logical gates that are more complex than traditional classical gates, such as **AND** and **OR**
  - **Pauli-X Gate** or the quantum NOT, **swaps** the amplitude of the **|0⟩ and |1⟩** components
    - Represented by the (insert here) matrix
  - **Hadamard Gate** creates a **superposition** from a |0⟩ or |1⟩
    - This is represented by **(1/√2)(|0⟩ + |1⟩)**
    - It is also **reversible**; applying an H Gate twice returns the initial input
- Grover's Algorithm achieves its $O(\sqrt{N})$ complexity through the use of **Hadamard Gates** and **modified Z Gates**.
  - This complexity is **"asymptotically optimal"**
    - So Grover's Algorithm is the best solution! (Fig. 1)

Conclusion:

- Classical searching algorithm can be implemented in quantum
- Encryption levels need to be increased within the next ten years
  - Especially for searching algorithms that have faster speedup (Shor's Algorithm)
- Grover's Algorithm for Sudoku.
  - Grover's algorithm searches all possible Sudoku answers efficiently.
  - Grover's algorithm speeds up search by reducing the number of checks needed to find the correct answer from N to about √N.

Process & Results:

Fig. 1:

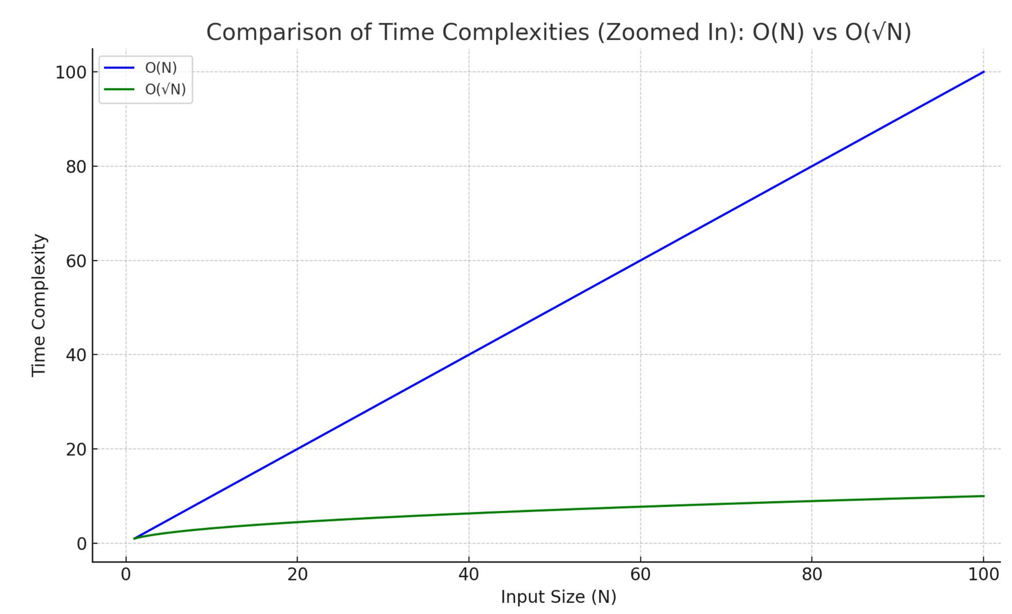

Fig. 2:

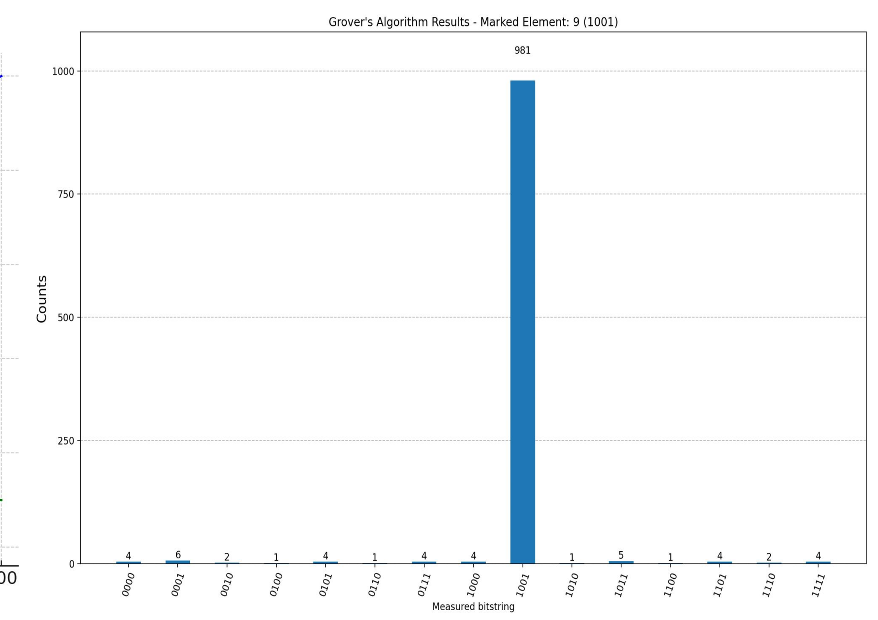

Fig. 5

```
puzzle = np.array([[0,0,3,0],
    [0,0,1,0],
    [0,0,0,1],
    [3,0,2,0]])

row = {'00000': [1,2,3,4],…
N=544935

def isSolution(N):
    bin_N= bin(N)[2:]
    padding = '0'*(20-len(bin_N))
    padded_bin_N = padding+bin_N
    #print(padded_bin_N)
    row1 = padded_bin_N[0:5]
    row2 = padded_bin_N[5:10]
    row3 = padded_bin_N[10:15]
    row4 = padded_bin_N[15:20]
    if int(row1,2) > 23:…
    if int(row2,2) > 23:…
    if int(row3,2) > 23:…
    if int(row4,2) > 23:
        return False
    trialSolution = np.array([row[row1], row[row2], row[row
    #print(puzzle)
    #print(trialSolution)
    match = (trialSolution == puzzle)
    sol_match = match[puzzle > 0]
    #print(sol_match)
    if sol_match.all() == False:…
    col1 = trialSolution[:,0]
    col2 = trialSolution[:,1]
    col3 = trialSolution[:,2]
    col4 = trialSolution[:,3]
    col1_sorted = np.sort(col1)
    compare_col1 = ( col1_sorted == np.array([1,2,3,4]) )
    #print(compare_col1.all())
    if compare_col1.all() == False:…
    col2_sorted = np.sort(col2)
```

Fig. 3:

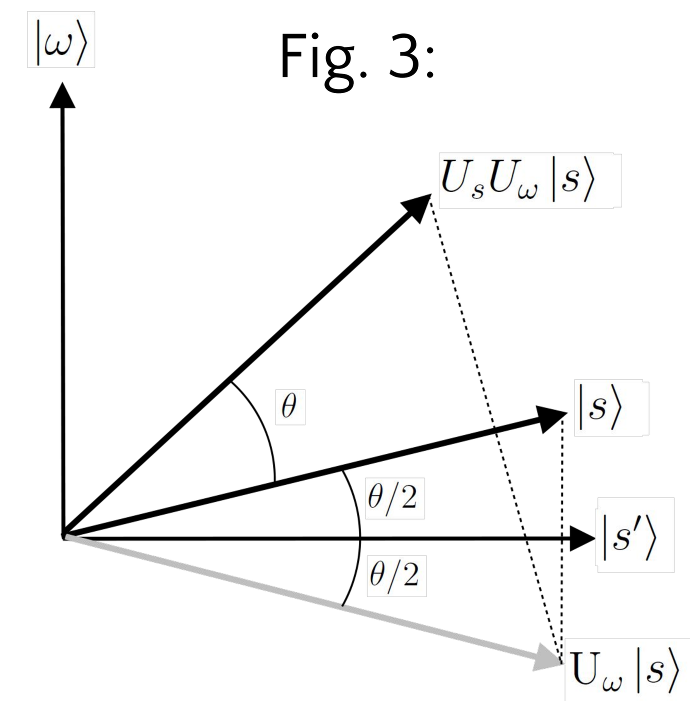

Fig. 4:

		3	
		1	
			1
3		2	

- Apply an **H Gate** to **n** qubits, giving an array with **2^n values**, all with an equal probability of being chosen

$$|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle.$$

- Iterate the following steps $\frac{\pi}{4}\sqrt{N}$ **times**
  - Apply an **Oracle** function that multiplies **only the solution state by -1**

$$\begin{cases} U_\omega|x\rangle = -|x\rangle & \text{for } x = \omega, \text{ that is, } f(x) = 1, \\ U_\omega|x\rangle = |x\rangle & \text{for } x \neq \omega, \text{ that is, } f(x) = 0. \end{cases}$$

  - **Amplify** the magnitude of the flipped state
    - This manifests as a **reflection** around the **previous vector**
    - Apply H and X gates to all qubits, apply a multi-controlled Z gate on target qubit, then undo the H and X gates
- After all iterations for large datasets, Grover's algorithm has a chance to succeed $P_{success} > 1 - (1/N)$
  - This chance becomes **extremely large** as N increases (Fig. 2)

Acknowledgements:

Enormous thanks to Skandaprasad Rao and the students at Harvard and MIT for their quantum expertise! Thank you to Mrs. Chopra and Dr. Newland for allowing us to use their rooms after school. Thank you to 3Blue1Brown for a comprehensive video lesson on Grover's algorithm.