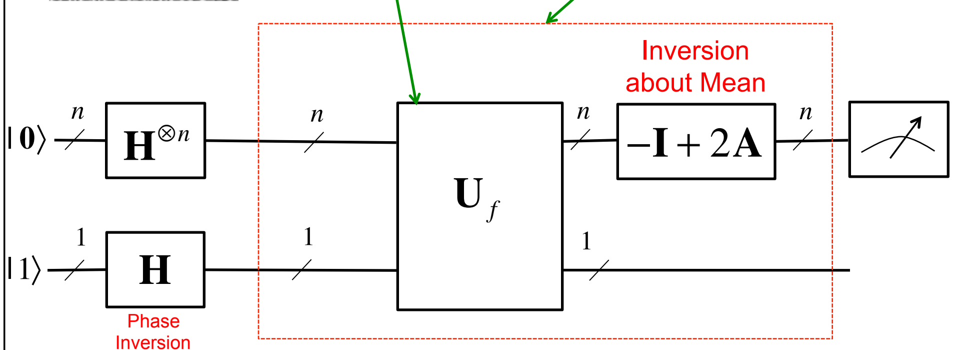
Grover Search Algorithm



Grover's Search Algorithm

- Method for Searching for Single Element in Among Array of N Elements
- Not Exponential Speedup Like Other Algorithms – Speedup is Quadratic

$$O\left(\sqrt{N}\right)$$

- Like Periodicity Algorithm, Method is Probabilistic
  - Requires Several Evaluations for Answer
- Cascade uses a Structure Known as Grover's Oracle
  - Yields "1" if Object Present and "0" if Not

Grover's Search Algorithm

- Cast Search Problem in terms of Searching for a Binary String $\mathbf{x}_0$
- We Utilize an Oracle Function of the Form:

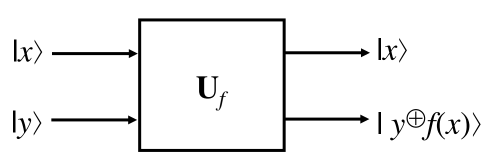
$$f(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} = \mathbf{x}_0 \\ 0, & \mathbf{x} \neq \mathbf{x}_0 \end{cases}$$

- Classical Algorithm would have to Evaluate all 2^n Strings (worst case)
- Grover's Method Requires: $\sqrt{2^n} = 2^{\frac{n}{2}}$

Specification of Function

- Specified as Unitary Operation that Performs the Transformation:

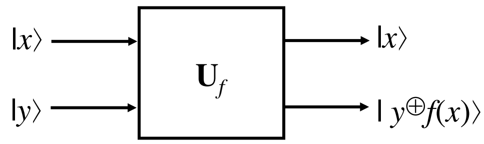
$$|\mathbf{x}, y\rangle \mapsto |\mathbf{x}, y \oplus f(\mathbf{x})\rangle$$



Function Example

- Consider $n=2$ Where $f(\mathbf{x})$ Detects the Bitstring $\mathbf{x}_0=10$:

$$|\mathbf{x},y\rangle \mapsto |\mathbf{x},y\oplus f(\mathbf{x})\rangle$$



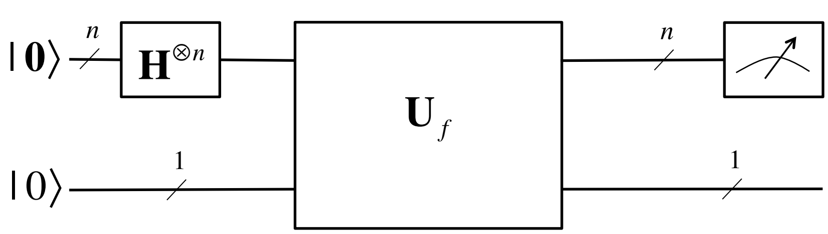
$$\mathbf{U}_f = |00,0\rangle\langle 00,0| + |00,1\rangle\langle 00,1| + |01,0\rangle\langle 01,0| + |01,1\rangle\langle 01,1| + |10,1\rangle\langle 10,0| + |10,0\rangle\langle 10,1| + |11,0\rangle\langle 11,0| + |11,1\rangle\langle 11,1|$$

$$\mathbf{U}_f = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

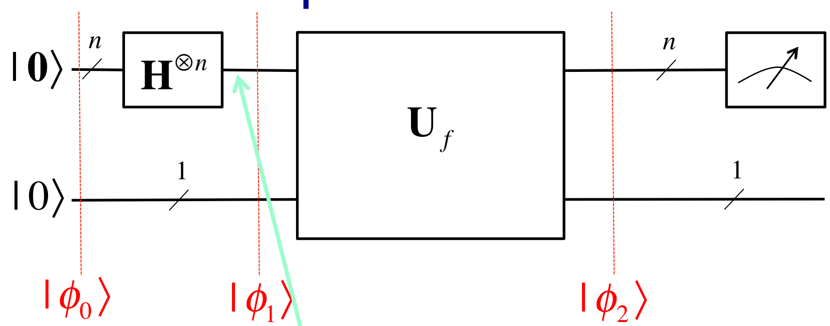
- Must consider ancilla equal to both $|0\rangle$ and $|1\rangle$ since H gate is used

First Attempt to Solve Problem

- Use Our "Favorite Trick" of Placing Function Input into State of Superposition
- Then, Perform a Measurement



First Attempt to Solve Problem



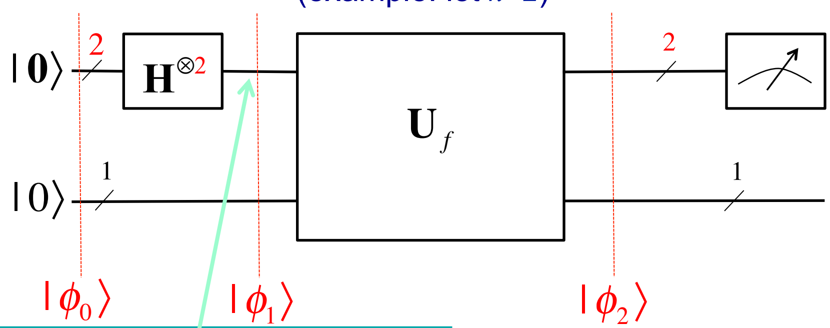
$$\left[\frac{\sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle}{\sqrt{2^n}}\right]|\mathbf{0}\rangle$$

$$|\phi_2\rangle = \left[\frac{\sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}, f(\mathbf{x})\rangle}{\sqrt{2^n}}\right]$$

$$|\phi_2\rangle = \mathbf{U}_f\left(\mathbf{H}^{\otimes n}\otimes \mathbf{I}_2\right)|\mathbf{0},0\rangle$$

First Attempt to Solve Problem

(example: let n=2)



$$\left[\frac{\sum_{\mathbf{x}\in\{0,1\}^2} |\mathbf{x}\rangle}{\sqrt{2^2}}\right]|00\rangle = \frac{1}{2}\left(|00\rangle+|01\rangle+|10\rangle+|11\rangle\right)$$

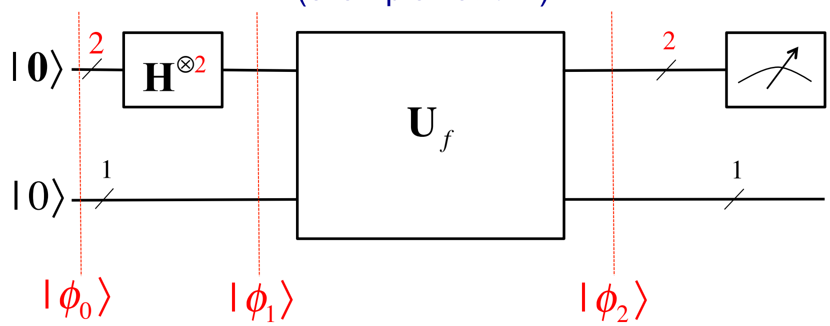
$$|\phi_1\rangle = \left[\frac{\sum_{\mathbf{x}\in\{0,1\}^2} |\mathbf{x},0\rangle}{\sqrt{2^2}}\right] = \frac{1}{2}\left(|00,0\rangle+|01,0\rangle+|10,0\rangle+|11,0\rangle\right)$$

$$|\phi_2\rangle = \left[\frac{\sum_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}, f(\mathbf{x})\rangle}{\sqrt{2^n}}\right]$$

$$|\phi_2\rangle = \mathbf{U}_f\left(\mathbf{H}^{\otimes n}\otimes \mathbf{I}_2\right)|\mathbf{0},0\rangle$$

First Attempt to Solve Problem

(example: let $n=2$)



$$|\phi_1\rangle = \left[\frac{\sum_{\mathbf{x}\in\{0,1\}^2} |\mathbf{x},0\rangle}{\sqrt{2^2}}\right] = \frac{1}{2}\left(|00,0\rangle + |01,0\rangle + |10,0\rangle + |11,0\rangle\right)$$

$$|\phi_2\rangle = \mathbf{U}_f\left(\mathbf{H}^{\otimes n} \otimes \mathbf{I}_2\right)|\mathbf{0},0\rangle$$

$$|\phi_2\rangle = \left[\frac{\sum_{\mathbf{x}\in\{0,1\}^2} |\mathbf{x},f(\mathbf{x})\rangle}{\sqrt{2^2}}\right] = \frac{1}{2}\left(|00,0\rangle + |01,0\rangle + |10,1\rangle + |11,0\rangle\right)$$

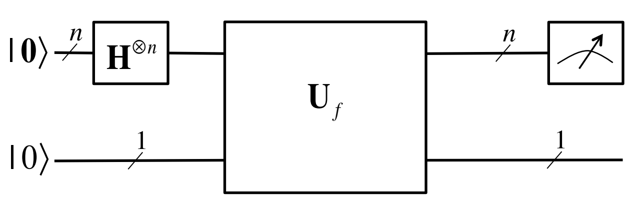
180° Phase Shift (inversion)
string is "marked"

First Attempt to Solve Problem

- Measuring Top n Qubits Yields one of 2^n Bitstrings with Equal Probability
- Measuring Bottom Qubit Yields:

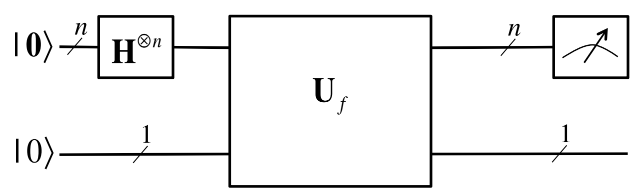
  $|0\rangle$ with probability $\frac{2^n-1}{2^n}$ $\qquad$ $|1\rangle$ with probability $\frac{1}{2^n}$

- If Lucky, Measure $|1\rangle$ and Top Qubits Yield Bitstring being Searched for Since they are Entangled with Bottom Bit

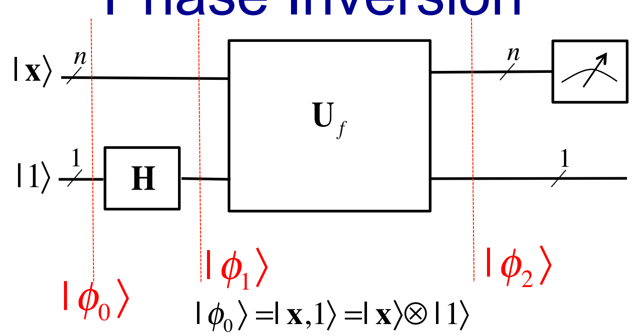


First Attempt to Solve Problem

- Since Chances of Measuring Desired Output are Small, Need Additional Operations
- Use Two New Tricks
  - Phase Inversion
  - Inversion About the Mean (or Average)



Phase Inversion



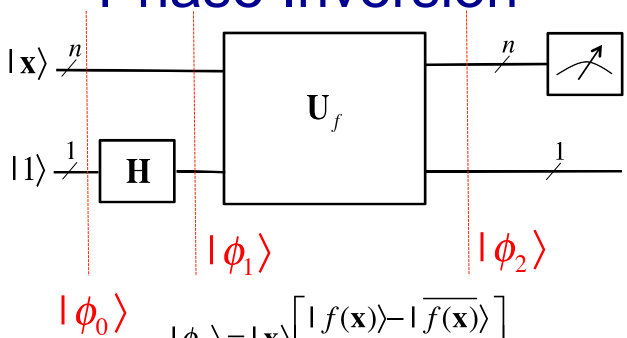
$$|\phi_0\rangle = |\mathbf{x},1\rangle = |\mathbf{x}\rangle \otimes |1\rangle$$

$$|\phi_1\rangle = \left(\mathbf{I}_n \otimes \mathbf{H}\right)|\mathbf{x},1\rangle = |\mathbf{x}\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \left[\frac{|\mathbf{x},0\rangle - |\mathbf{x},1\rangle}{\sqrt{2}}\right]$$

$$|\phi_2\rangle = \left(\mathbf{I}_n \otimes \mathbf{H}\right)\left(\mathbf{U}_f\right)|\mathbf{x},1\rangle = |\mathbf{x}\rangle\left[\frac{|f(\mathbf{x})\oplus 0\rangle - |f(\mathbf{x})\oplus 1\rangle}{\sqrt{2}}\right]$$

$$= |\mathbf{x}\rangle\left[\frac{|f(\mathbf{x})\rangle - |\overline{f(\mathbf{x})}\rangle}{\sqrt{2}}\right]$$

Phase Inversion



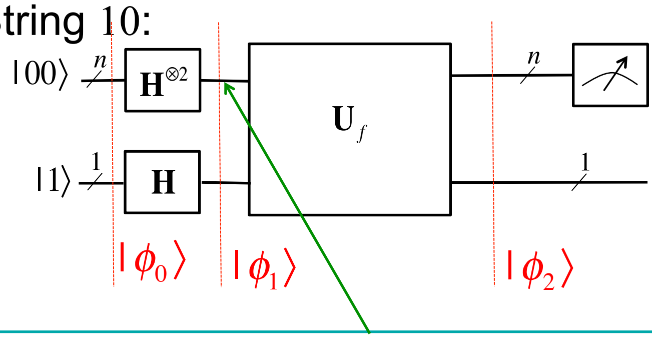
$$|\phi_2\rangle = |\mathbf{x}\rangle\left[\frac{|f(\mathbf{x})\rangle - |\overline{f(\mathbf{x})}\rangle}{\sqrt{2}}\right]$$

- Since a-b=(-1)(b-a):

$$|\phi_2\rangle = (-1)^{f(x)}|\mathbf{x}\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \begin{cases} (-1)|\mathbf{x}\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], & \text{if } \mathbf{x} = \mathbf{x}_0 \\ (+1)|\mathbf{x}\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right], & \text{if } \mathbf{x} \neq \mathbf{x}_0 \end{cases}$$

Phase Inversion Example

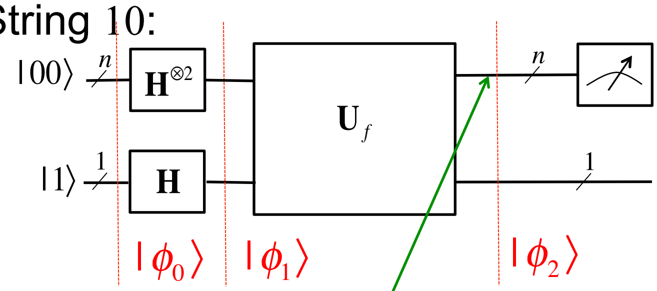
- Assume n=2 and Top n Qubits are in Equal State of Superposition and $f(\mathbf{x})$ "Chooses" the String 10:



$$(\mathbf{H}\otimes\mathbf{H})|00\rangle = \frac{1}{\sqrt{2^2}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{bmatrix}$$

Phase Inversion Example

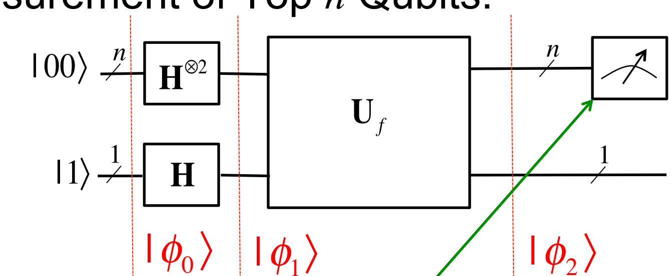
- Assume n=2 and Top n Qubits are in Equal State of Superposition and $f(\mathbf{x})$ “Chooses” the String 10:



$$(\mathbf{H}\otimes\mathbf{H})(\mathbf{U}_f)|00\rangle = \frac{1}{\sqrt{2^2}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}$$

Phase Inversion Example

- Measurement of Top n Qubits:

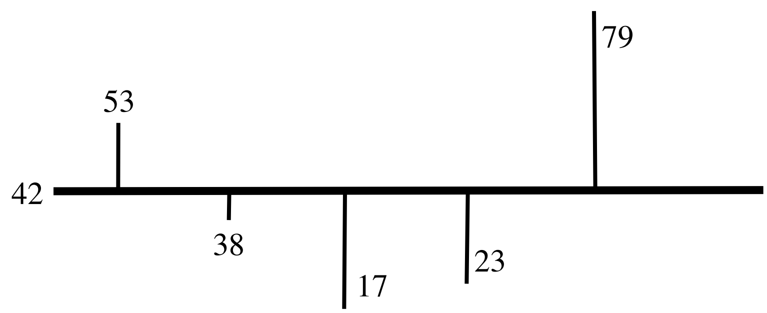


$$\text{Measure}\begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}; \quad \text{Prob}(|\mathbf{x}\rangle = 00,01,10,11) = \begin{cases} (1/2)^2, & \mathbf{x} = 00,01,11 \\ (-1/2)^2 & \mathbf{x} = 10 \end{cases} = \frac{1}{4}$$

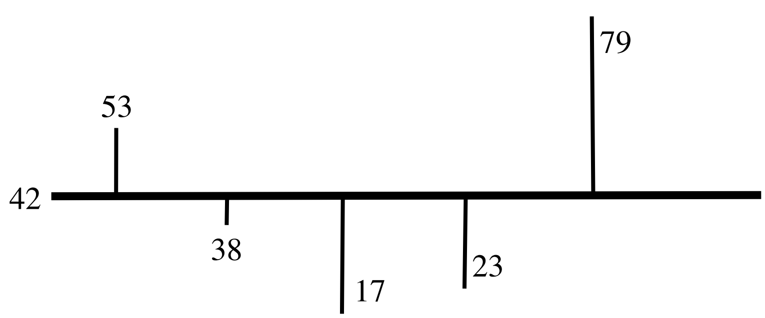
- Need Another “Trick” – Inversion About Mean:

Inversion About the Mean

- Must “Boost” Phase Separation Among Bitstrings
- Use “Inversion About the Mean”
- Consider a Set of Values {53,38,17,23,79}:
- Average{53,38,17,23,79}=42



Inversion About the Mean



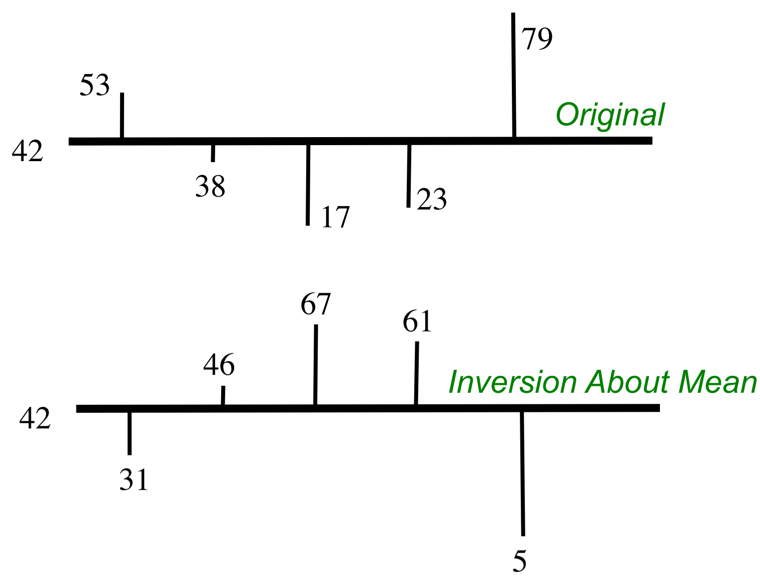
- Desired to Change Sequence so That Each Element Above Average is Same Distance from Average but Below
- Each Element Below Average is Same Distance from Average but Above

Inversion About the Mean

- To Do This, We INVERT Each Element About Average
  - Move Those Above to Below and vice versa
- EXAMPLE: First Element is 53 and AVG-53=42-53=-11 so 11 Units Below (deviation)
- Add AVG=42 to Deviation (-11)
- Obtain AVG+(AVG-53)=42+(42-52)=31
- Second Element Becomes 42+(42-38)=46
- Each Element v Changed to v' :

$$v' = \mathrm{AVG} + (\mathrm{AVG} - v)$$
$$= 2(\mathrm{AVG}) - v$$

Inversion About the Mean



Inversion About the Mean

- Formulate Inversion About Mean as Matrix Operation
- Consider the Set of Values $\{53,38,17,23,79\}$:
- We Write the Set as a Column Vector and use an Averaging Matrix, $\mathbf{A}$:

$$average\{53,38,17,23,79\} = \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \end{bmatrix} \begin{bmatrix} 53 \\ 38 \\ 17 \\ 23 \\ 79 \end{bmatrix}$$

- Vector of values is $\mathbf{v}^T=[53\ \ 38\ \ 17\ \ 23\ \ 79]$
- Vector of averages is $(\mathbf{Av})^T=[42\ 42\ 42\ 42\ 42]$

$$v' = 2(\text{AVG}) - v \qquad \mathbf{v}' = 2\mathbf{Av} - \mathbf{v} = (2\mathbf{A} - \mathbf{I})\mathbf{v}$$

Inversion About the Mean

$$average\{53,38,17,23,79\} = \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \end{bmatrix} \begin{bmatrix} 53 \\ 38 \\ 17 \\ 23 \\ 79 \end{bmatrix}$$

$$average\{53,38,17,23,79\} = \mathbf{A}_5\mathbf{v} = \frac{53+38+17+23+79}{5} = 42$$

$$\mathbf{v}' = -\mathbf{v} + 2\mathbf{A}_n\mathbf{v} = (-\mathbf{I}_n + 2\mathbf{A}_n)\mathbf{v}$$

Inversion About the Mean

$$\mathbf{A}_5 = \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} 53 \\ 38 \\ 17 \\ 23 \\ 79 \end{bmatrix}$$

$$2\mathbf{A}_5 = \begin{bmatrix} 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \end{bmatrix} \qquad \mathbf{I}_5 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{v}' = -\mathbf{v} + 2\mathbf{A}_n\mathbf{v} = (-\mathbf{I}_n + 2\mathbf{A}_n)\mathbf{v}$$

Inversion About the Mean

$$-\mathbf{I}_5 + 2\mathbf{A}_5 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} + \begin{bmatrix} 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & 2/5 \end{bmatrix}$$

$$-\mathbf{I}_5 + 2\mathbf{A}_5 = \begin{bmatrix} (-1+2/5) & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & (-1+2/5) & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & (-1+2/5) & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & (-1+2/5) & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & (-1+2/5) \end{bmatrix}$$

$$\mathbf{v}' = -\mathbf{v} + 2\mathbf{A}_n\mathbf{v} = (-\mathbf{I}_n + 2\mathbf{A}_n)\mathbf{v}$$

Inversion About the Mean

$$\mathbf{v}' = -\mathbf{v} + 2\mathbf{A}_n\mathbf{v} = (-\mathbf{I}_n + 2\mathbf{A}_n)\mathbf{v}$$

$$(-\mathbf{I}_5 + 2\mathbf{A}_5)\mathbf{v} = \begin{bmatrix} (-1+2/5) & 2/5 & 2/5 & 2/5 & 2/5 \\ 2/5 & (-1+2/5) & 2/5 & 2/5 & 2/5 \\ 2/5 & 2/5 & (-1+2/5) & 2/5 & 2/5 \\ 2/5 & 2/5 & 2/5 & (-1+2/5) & 2/5 \\ 2/5 & 2/5 & 2/5 & 2/5 & (-1+2/5) \end{bmatrix} \begin{bmatrix} 53 \\ 38 \\ 17 \\ 23 \\ 79 \end{bmatrix} = \begin{bmatrix} 31 \\ 46 \\ 67 \\ 61 \\ 5 \end{bmatrix}$$

- To Generalize, Consider n Qubits
- Quantum State Vector Contains 2^n Elements
- Form of $\mathbf{A}$ Matrix is:

$$\mathbf{A}_n = \left[1/2^n \right]_{n\times n}$$

Inversion About the Mean Property

$$-\mathbf{I}_n + 2\mathbf{A}_n = \begin{bmatrix} (-1+2/2^n) & 2/2^n & \dots & 2/2^n \\ 2/2^n & (-1+2/2^n) & \dots & 2/2^n \\ \vdots & \vdots & \ddots & \vdots \\ 2/2^n & 2/2^n & \dots & (-1+2/2^n) \end{bmatrix}$$

$$\mathbf{B}_n = \mathbf{A}_n^2 = \left[b_{ij} \right]_{n\times n}$$

$$b_{ij} = \sum_{k=1}^{2^n} (1/2^n)^2 = (2^n)(1/2^n)^2 = (2^n)(1/2^{2n}) = \frac{2^n}{2^{2n}} = \frac{2^n/2^n}{2^{2n}/2^n} = \frac{1}{2^{2n-n}} = \frac{1}{2^n}$$

$$\mathbf{A}_n^2 = \mathbf{A}_n \times \mathbf{A}_n = \mathbf{A}_n$$

$$(-\mathbf{I}_n + 2\mathbf{A}_n)(-\mathbf{I}_n + 2\mathbf{A}_n) = \mathbf{I}_n - 2\mathbf{A}_n - 2\mathbf{A}_n + 4\mathbf{A}_n^2$$

$$(-\mathbf{I}_n + 2\mathbf{A}_n)(-\mathbf{I}_n + 2\mathbf{A}_n) = \mathbf{I}_n - 2\mathbf{A}_n - 2\mathbf{A}_n + 4\mathbf{A}_n$$

$$(-\mathbf{I}_n + 2\mathbf{A}_n)(-\mathbf{I}_n + 2\mathbf{A}_n) = \mathbf{I}_n$$

Unitary Operation!!!!! Realizable as Quantum Operation!!!

Inversion About the Mean
Previous 2 Qubit Example

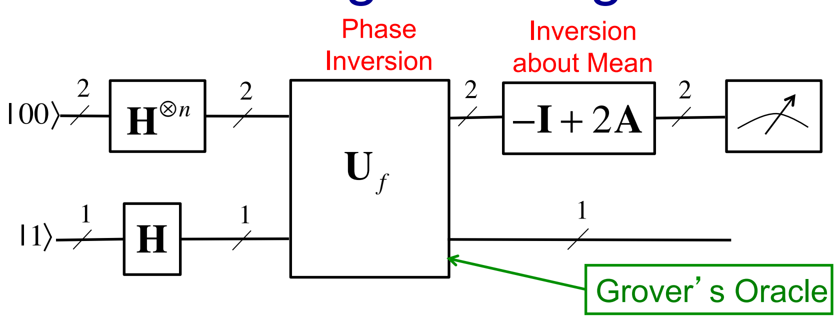
- After Phase Inversion:

$$\mathbf{v} = \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}$$

$$-\mathbf{I}_4 + 2\mathbf{A}_4 = \begin{bmatrix} (-1+2/4) & 2/4 & 2/4 & 2/4 \\ 2/4 & (-1+2/4) & 2/4 & 2/4 \\ 2/4 & 2/4 & (-1+2/4) & 2/4 \\ 2/4 & 2/4 & 2/4 & (-1+2/4) \end{bmatrix}$$

$$(-\mathbf{I}_4 + 2\mathbf{A}_4)\mathbf{v} = \begin{bmatrix} (-1+2/4) & 2/4 & 2/4 & 2/4 \\ 2/4 & (-1+2/4) & 2/4 & 2/4 \\ 2/4 & 2/4 & (-1+2/4) & 2/4 \\ 2/4 & 2/4 & 2/4 & (-1+2/4) \end{bmatrix} \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Putting it All Together



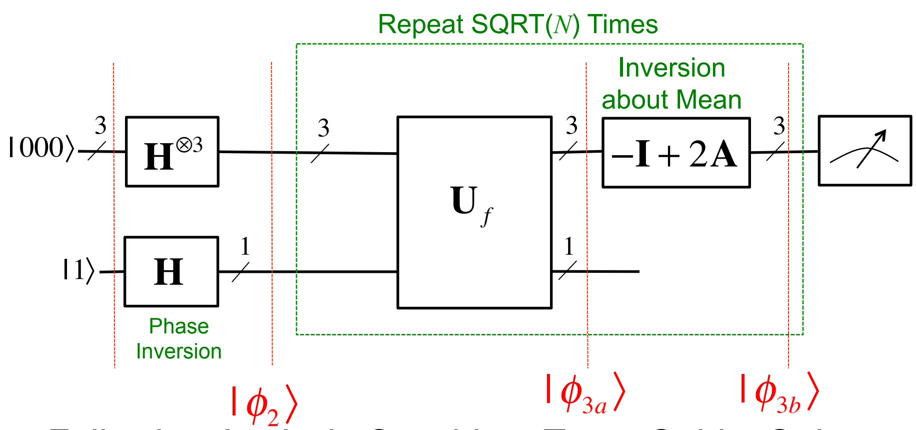
- Classical Computer Search Requires Evaluation of $f(\mathbf{x})$ $4=2^2$ Times
- Quantum Search Requires One Evaluation of Oracle
- For Larger Problems Need to Evaluate Oracle $\sqrt{N} = \sqrt{2^n}$ Times

Larger Search Problem

- Consider Search Problem for Bitstring of Length 3
- Oracle Unitary Operation Embeds Boolean Function that Produces a 1 When String 101 is Domain Argument and 0 Otherwise

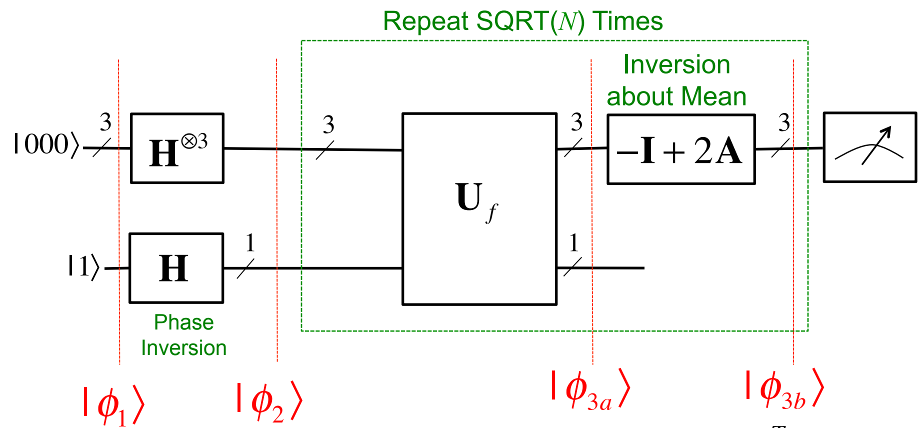
$$f(\mathbf{x}) = f(x_1 x_2 x_3) = \begin{cases} 1, & \mathbf{x} = 101 \\ 0, & \mathbf{x} \neq \mathbf{x}_0 \end{cases}$$

Search Problem for n=3



- Following Analysis Considers Top 3 Qubits Only
- Must Cascade Portion in Green Several Times to Enhance Effect of Inversion About Mean
- Example Illustrates this Process

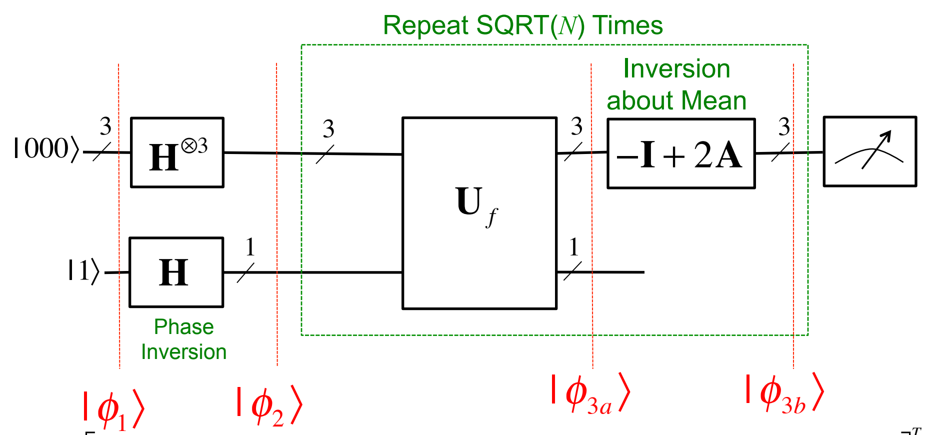
Search Problem for n=3



$$|\phi_1\rangle = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

$$|\phi_2\rangle = \begin{bmatrix} 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} \end{bmatrix}^T$$

Search Problem for n=3



$$|\phi_{3a}\rangle = \begin{bmatrix} 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} & -1/\sqrt{2^3} & 1/\sqrt{2^3} & 1/\sqrt{2^3} \end{bmatrix}^T$$

000 001 010 011 100 101 110 111

Search Problem for $n=3$

$$|\phi_{3a}\rangle = \begin{bmatrix} \overset{000}{1/\sqrt{2^3}} & \overset{001}{1/\sqrt{2^3}} & \overset{010}{1/\sqrt{2^3}} & \overset{011}{1/\sqrt{2^3}} & \overset{100}{1/\sqrt{2^3}} & \overset{101}{-1/\sqrt{2^3}} & \overset{110}{1/\sqrt{2^3}} & \overset{111}{1/\sqrt{2^3}} \end{bmatrix}^T$$

- Calculating the Average of These Values:

$$average = a = \frac{7\times\frac{1}{\sqrt{8}} - \frac{1}{\sqrt{8}}}{8} = \frac{3}{4\sqrt{8}}$$

- Inversion about Mean for $i=\{000,001,010,011,100,110,111\}$:

$$-v_i + 2a = -\frac{1}{\sqrt{8}} + \left(2\times\frac{3}{4\sqrt{8}}\right) = \frac{1}{2\sqrt{8}}$$

- Inversion about Mean for 101:

$$-v_i + 2a = \frac{1}{\sqrt{8}} + \left(2\times\frac{3}{4\sqrt{8}}\right) = \frac{5}{2\sqrt{8}}$$

Search Problem for $n=3$

$$|\phi_{3b}\rangle = \begin{bmatrix} \overset{000}{1/2\sqrt{8}} & \overset{001}{1/2\sqrt{8}} & \overset{010}{1/2\sqrt{8}} & \overset{011}{1/2\sqrt{8}} & \overset{100}{1/2\sqrt{8}} & \overset{101}{5/2\sqrt{8}} & \overset{110}{1/2\sqrt{8}} & \overset{111}{1/2\sqrt{8}} \end{bmatrix}^T$$

- If Measurement Performed Now, Probability of Finding the Search Bitstring is:

$$\text{Prob}\left[|\phi_{3b}\rangle = |101\rangle\right] = \left(5/2\sqrt{8}\right)^2 = \frac{25}{32} = 0.78$$

- If Measurement Performed Now, Probability of Finding One of 7 Other Bitstrings is:

$$\text{Prob}\left[|\phi_{3b}\rangle \neq |101\rangle\right] = 7\times\left(1/2\sqrt{8}\right)^2 = \frac{7}{32} = 0.22$$

Search Problem for n=3

- Desirable to Increase Probability of Measuring the Bitstring we are Searching For
- To Do This, We Phase Invert and Invert About Mean Again
- Implemented By Cascading Green Boxes
- Next Phase Inversion Yields:

$$
\begin{array}{lcccccccc}
 & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\
|\phi_{3c}\rangle = \Big[& \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{-5}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} & \frac{1}{2\sqrt{8}} \Big]^T
\end{array}
$$

$$average = a = \frac{7\times\frac{1}{2\sqrt{8}} - \frac{5}{2\sqrt{8}}}{8} = \frac{1}{8\sqrt{8}}$$

Search Problem for n=3

- Inverting About the Mean Yields:

$$-v_i + 2a = -\frac{1}{2\sqrt{8}} + \left(2\times\frac{1}{8\sqrt{8}}\right) = -\frac{1}{4\sqrt{8}} \qquad -v_i + 2a = \frac{5}{2\sqrt{8}} + \left(2\times\frac{1}{4\sqrt{8}}\right) = \frac{11}{2\sqrt{8}}$$

$$
\begin{array}{lcccccccc}
 & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\
|\phi_{3d}\rangle = \Big[& \frac{-1}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} & \frac{11}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} & \frac{-1}{4\sqrt{8}} \Big]^T
\end{array}
$$

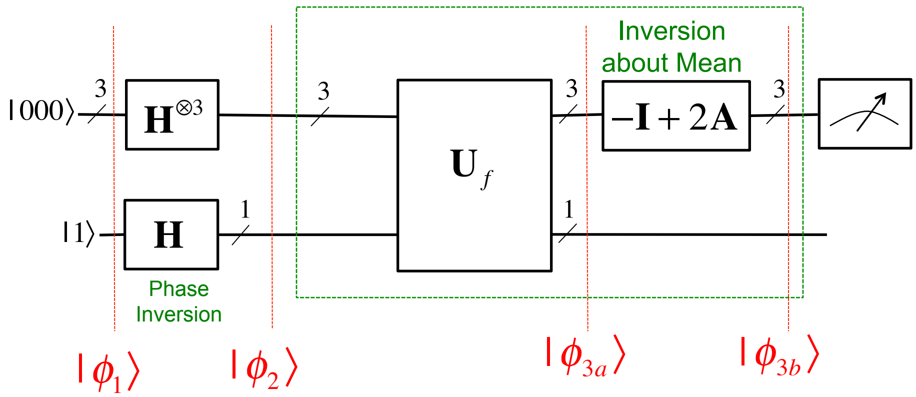
- If Measurement Performed Now, Probability of Finding the Search Bitstring is:

$$\text{Prob}\left[|\phi_{3d}\rangle = |101\rangle\right] = \left(\frac{11}{4\sqrt{8}}\right)^2 = \frac{121}{128} = 0.95$$

- If Measurement Performed Now, Probability of Finding One of 7 Other Bitstrings is:

$$\text{Prob}\left[|\phi_{3d}\rangle \neq |101\rangle\right] = 7\times\left(\frac{1}{4\sqrt{8}}\right)^2 = \frac{7}{128} = 0.05$$

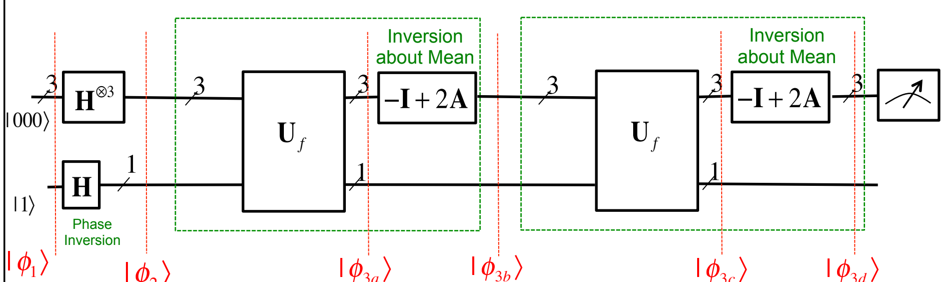
Search Problem for n=3



$$\text{Prob}\left[|\phi_{3b}\rangle = |101\rangle\right] = \left(\frac{5}{2\sqrt{8}}\right)^2 = \frac{25}{32} = 0.78$$

$$\text{Prob}\left[|\phi_{3b}\rangle \neq |101\rangle\right] = 7 \times \left(\frac{1}{2\sqrt{8}}\right)^2 = \frac{7}{32} = 0.22$$

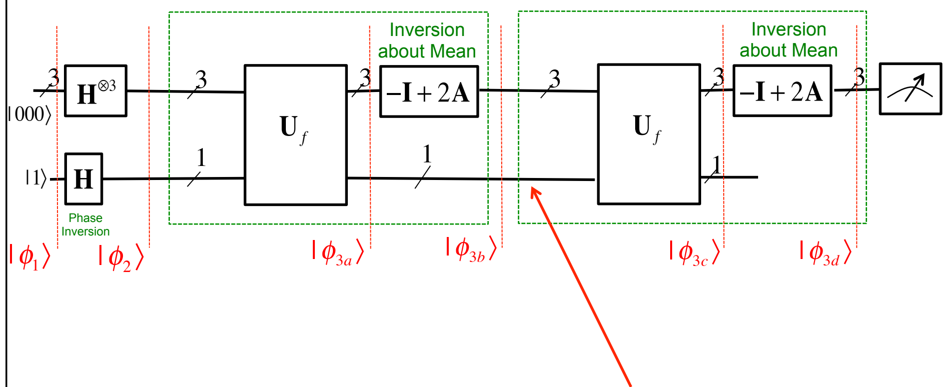
Search Problem for n=3



$$\text{Prob}\left[|\phi_{3d}\rangle = |101\rangle\right] = \left(\frac{11}{4\sqrt{8}}\right)^2 = \frac{121}{128} = 0.95$$

$$\text{Prob}\left[|\phi_{3d}\rangle \neq |101\rangle\right] = 7 \times \left(\frac{1}{4\sqrt{8}}\right)^2 = \frac{7}{128} = 0.05$$

Search Problem for n=3



No Need to Repeat Single-qubit Hadamard at Lower Part Phase Inversion of Marking of Desired String Only Needed the First time

Grover's Search Algorithm

- Probabilistic Algorithm
- Need to Repeat "Green Box" SQRT(N) Times
  - SQRT(N)=SQRT(2^n) in this Example
- Quadratic Speedup Since Classical Computer Requires N Evaluations and Quantum Computer (implementing Grover's Method) Requires SQRT(N)
- Can Generalize, Search for t Elements Instead of 1 Element Requires SQRT(N/t) "Green Boxes"

Grover' s Search Algorithm

- Some Literature Considers "Green Box" to be the "Oracle" – Other Considers the Unitary Operation $\mathbf{U}_f$ to be "Oracle"
- Many Problems can be Formulated as Search Problems – Offers Quadratic Speedup
- Must Determine the Oracle – this is the Challenge
- Unlike Other Quantum Algorithms, Grover' s Method DOES NOT Provide Exponential Speedup

Grover' s Search Algorithm

- Importance of Grover's Search is that a QUERY is Accomplished with Quadratic Speedup
- If Oracle Requires Searching through all Strings then no Performance Gain
- Many Modified Versions
- Also, Adaptations to Represent Data in a Quantum Form if it Does NOT Contain All Possible Elements
- Main Contributions: Amplitude Amplification through Inversion about Mean, AND, Phase Inversion