Grover's Algorithm
Quantum Parallelism

Joseph Spring

Applications
Outline

1. Introduction
2. The Search Algorithm
   - One or Two Points
3. Grovers Algorithm
   - The Algorithm
   - Grover Iteration
   - Findings
4. References
We have discussed the concept of quantum parallelism and now consider a range of applications. These will include:

- **The Deutsch-Josza Algorithm**
  - One of the first of this type of algorithm suggesting potential benefits from quantum algorithms based on quantum parallelism and interference over classical algorithms.

- **Shor’s Algorithm**
  - This has the potential to make any public key cryptography algorithm based on either the DLP or the IFP redundant. No more RSA, El-Gamal, ...

- **Grovers Algorithm**
  - A search algorithm offering potential benefits over classical alternatives. Is this the end of symmetric key cryptography?
Grover's algorithm (1996) is a quantum algorithm designed to search an unsorted database containing N entries, for a specific unique item. It runs in a time of the order $O(\sqrt{N})$ and uses $O(\log N)$ space.

Like Shor’s Algorithm it is a probabilistic with high probability of producing the correct answer.

It is said to offer quadratic speedup over corresponding classical search algorithms.

- Note this is not exponential speedup
- It is still a considerable improvement on alternative algorithms gaining significant advantage as the size N of the database increases

If there exists M copies of the item being searched for then the search time is of the order $O(\sqrt{\frac{N}{M}})$.
A search algorithm is a procedure for finding an element that satisfies given criteria, from a collection of given elements. The elements may be part of a database (a collection of records/items), a search space (defined by a mathematical formula or procedure) or a combination of the two (Hamiltonian path in graph theory - visiting each vertex in a graph once only)

- Classical Search Algorithm
  - Linear/Sequential Search. Start at beginning and compare each entry to criteria. Average Time $O(N)$
  - Binary Search (sorted database), start in middle, compare to criteria, work with upper half or lower half until item found. Average Time $O(logN)$
  - Travelling Salesman: (Shortest Route), $O(N)$, Backtracking, Alpha-Beta Minimax Search (Chess Games), . . .
Search Algorithm

Procedure

- Let $S = \{ e_i \}_{i=0}^{N=2^n-1}$ denote the search space.
- Let $M \leq N$ denote the number of elements that satisfy the query.
- The Problem: Identify one element satisfying the query.
- Procedure:
  - Select an element $e_i$ of $S$ and determine whether or not the element satisfies the query.
  - Let $f(x)$ denote a function with $0 \leq x \leq N = 2^n - 1$ such that
    
    \[ f(x) = \begin{cases} 
    0 & \text{if } x \text{ does NOT satisfy the query} \\
    1 & \text{if } x \text{ does satisfy the query} 
    \end{cases} \]
The Search Algorithm

The Oracle $O$

- This is a ‘Black Box’ accepting an n-qubit input (indexed by $i$ for element $e_i$)
- The Oracle another input referred to as an ‘oracle qubit’. This is set to $|q\rangle$ and flipped to $|\bar{q}\rangle$ if the oracle recognises a solution. $q \in \{0, 1\}$
- The black box performs the following transformation

$$|i\rangle|q\rangle \xrightarrow{O} |q \oplus f(i)\rangle$$
Search Algorithm
The Oracle $O$

Suppose that the oracle qubit is set to $|−\rangle$. Then:

$$|i\rangle|−\rangle = \frac{1}{\sqrt{2}}|i\rangle(|0\rangle − |1\rangle)$$

$$O \xrightarrow{\bigg\{ \begin{align*}
\frac{1}{\sqrt{2}}|i\rangle(|0\rangle − |1\rangle) & \quad \text{if } f(i) = 0, \text{ so } e_i \text{ is not a solution} \\
−\frac{1}{\sqrt{2}}|i\rangle(|0\rangle − |1\rangle) & \quad \text{if } f(i) = 1, \text{ so } e_i \text{ is a solution}
\end{align*} \bigg\}} = (−1)^{f(i)}|i\rangle|−\rangle$$

Since the oracle qubit does not change overall, we describe the transformation $O$ as:

$$|i\rangle \xrightarrow{O} (−1)^{f(i)}|i\rangle$$
Grover's Algorithm

Grover's Algorithm involves the following steps:

1. Initialise the n qubits to $|0000\ldots00\rangle$
2. Apply Hadamard $H^n$
3. Apply the Grover iteration $O(\sqrt{N})$ times
   - Step a) Apply the Oracle
   - Step b) Apply Hadamard $H^n$ to the output
   - Step c) Perform Conditional Phase Shift
     $$ |i\rangle \mapsto (-1)^{f(i)} |j\rangle $$
   - Step d) Apply Hadamard $H^n$ to the output
4. Perform Measurement
Grover's Algorithm
Steps 1 and 2

Step 1 $|\psi_0\rangle = |0000\ldots000\rangle$

Step 2 $|\psi_1\rangle = H^n|0000\ldots000\rangle$

$\quad = H|0\rangle H|0\rangle H|0\rangle \ldots H|0\rangle$

$\quad = |+\rangle |+\rangle |+\rangle \ldots |+\rangle = \frac{1}{2^{n/2}} \sum_z |z\rangle$

For general $|\psi\rangle = |x_1 x_2 x_3 \ldots x_n\rangle$

$|\psi\rangle = H^n |x_1 x_2 x_3 \ldots x_n\rangle = H |x_1\rangle H |x_2\rangle H |x_3\rangle \ldots H |x_n\rangle$

$\quad = \frac{1}{2^{n/2}} \sum_z (-1)^{\langle x|z\rangle} |z\rangle$  (Lemma 3, Deutsch-Josza slides)
Outline

1 Introduction
2 The Search Algorithm
   • One or Two Points
3 Grovers Algorithm
   • The Algorithm
   • Grover Iteration
   • Findings
4 References
Grovers Algorithm

Circuit 1 - One Round of the Grover Iteration G

Joseph Spring

Grovers Algorithm
We note that:

- The Phase Shift Gate leaves $|i\rangle$ alone if $f(i) = 0$ and transforms $|i\rangle$ to $-|i\rangle$ when $f(i) > 0$
- It may be shown that $G = (2|\psi\rangle\langle\psi| - I)O = f_{|\psi\rangle}O$

**Lemma**

Let $|\phi\rangle = \sum_{k=0}^{N-1} \alpha_k |k\rangle$

Then $f_{|\psi_1\rangle}|\phi\rangle = \sum_{r=0}^{N-1} \beta_r |r\rangle$ with $\beta_r = 2\alpha - \alpha_r$ and $\alpha = \frac{1}{N} \sum_{k=0}^{N-1} \alpha_k$
Step 3 - Grover Iteration

Circuit
Outline

1 Introduction

2 The Search Algorithm
   - One or Two Points

3 Grovers Algorithm
   - The Algorithm
   - Grover Iteration
   - Findings

4 References
The discussion on geometric interpretation of Grover’s algorithm in (for example) [2], [4] (or wikipedia) motivates the idea that repeated use of the Grover’s Iteration $G$, brings the vector $G|\psi\rangle$, ($|\psi\rangle$ denoting the initial state), closer and closer to $|b\rangle$, the state corresponding to a uniform superposition of all solutions to the query/search problem. This produces, with high probability, one of the outcomes superposed in $|b\rangle$.

$O(\sqrt{\frac{N}{M}})$ iterations are required to rotate $|\psi\rangle$ close to $|b\rangle$, explaining the efficiency of the algorithm.
Question: So what impact will this have on the demise, if any of symmetric key cryptography?

Conjecture: The speed up obtained is not exponential and so symmetric key problems remain outside of P complexity.

Comments?
References


