APPLYING QUANTUM ALGORITHM TO SPEED UP THE SOLUTION OF HAMILTONIAN CYCLE PROBLEMS

VIDYA RAJ C.
Assistant Professor
Dept. of Computer Science & Engineering
The National Institute of Engineering
Mysore – 570008, Karnataka, India

DR. M.S. SHIVAKUMAR
Principal
The National Institute of Engineering
Mysore – 570008, Karnataka, India

Abstract: Quantum computing is an important field of research that applies concepts of quantum physics to building more efficient computers. Although only rudimentary quantum computers have been built so far, many researchers believe that quantum computing has great potential and the quantum computers can efficiently perform some tasks which are otherwise not feasible on a classical computer. The Hamiltonian cycle problem is to determine whether a given graph has a Hamiltonian cycle or not. This problem belongs to the class of NP-complete problems, widely believed to intractable or hard on classical computers. Design of faster-than-classical quantum algorithms for important algorithmic problems has been an interesting intellectual adventure and achievement all along and their existence keeps being one of the key stimuli to those trying to overcome enormous technology problems to build (powerful) quantum computers. In this paper, we have used undirected graphs with varied number of vertices and we have shown how to determine the existence of a Hamiltonian cycle in a given graph. We have also illustrated how quantum search can be applied to obtain the solution of the Hamiltonian cycle problem much faster than the classical approach.

Key words: quantum computers, quantum algorithm, qubit, Hamiltonian cycle

1. INTRODUCTION

The current drive towards increasing speed and miniaturization of computers leads modern technology towards the subatomic domain - quantum computing - where strange quantum behavior takes over from familiar classical notions. Quantum computation touches upon the foundations of computer science, since quantum computers appear to violate the modern Church-Turing thesis [7]. Quantum computers can perform certain hard tasks, much faster than classical computers.

1.1 Quantum Computation

Quantum computing is a new, more powerful model of computing based on quantum mechanics. The basic variable used in quantum computing is a qubit, represented as a vector in a two dimensional complex Hilbert space where $|0>$ and $|1>$ form a basis in the space. The difference between qubits and bits is that a qubit can be in a state other than $|0>$ or $|1>$ whereas a bit has only one state, either 0 or 1. It is also possible to form linear combination of states, often called superposition.

The state of a qubit can be described by $|\Psi> = \alpha|0> + \beta|1>$ where, the numbers α and β are complex numbers. The special states $|0>$ and $|1>$ are known as computational basis states. We can examine a bit to determine whether it is in the state 0 or 1 but we cannot directly examine a qubit to determine its quantum state, that is values of α and β. When we measure a qubit we get either the result 0, with probability $|\alpha|^2$ or the result 1, with probability $|\beta|^2$, where $|\alpha|^2 + |\beta|^2 = 1$, since the probabilities must sum to one.

Consider the case of two qubits. In two classical bits there would be four possible states, 00, 01, 10 and 11. Correspondingly, a two qubit system has four computational basis states denoted $|00>$, $|01>$, $|10>$ and $|11>$. A pair of qubits can also exist in a superposition of these four states, so the quantum state of two qubits involves associating a complex coefficient, sometimes called amplitude, with each computational basis state, which is given as

$$|\Psi> = \alpha_{00}|00> + \alpha_{01}|01> + \alpha_{10}|10> + \alpha_{11}|11>$$

The logic that can be implemented with qubits is quite distinct from Boolean logic, and this is what has made quantum computing exciting by opening new possibilities [8].

1.2 Quantum Algorithms

Quantum Algorithms introduce a new paradigm of computation such as quantum superposition, quantum entanglement and promises to provide results that cannot be achieved by classical computers. Shor's factoring algorithm and Grover's search algorithm are examples of new algorithms that provide tremendous speedups over their classical counterparts [1][2][6].

In this paper, we would like to use Grover's search algorithm that can provide quadratic speedup, which is considerable when N is large over their classical counterparts.

Suppose we are given a map containing many cities, and wish to determine the shortest route passing through all the cities on the map. A simple algorithm to find this route is to search all possible routes through the cities, keeping a running record of which route has the shortest length. On a classical computer, if there are N possible routes, it takes O(N) operations to determine the shortest route using this method. But quantum search algorithm enables this search method to be sped up substantially, requiring only $O(\surd(N))$ operations[9][10]. The quantum search algorithm in general can be applied far beyond the route finding example just described to speed up many (though not all) classical algorithms that use search heuristics. Thus given a search space of size N, and no prior knowledge about the structure of information in it, if we want to find an element of search space satisfying a known property, then this problem requires approximately N operations, but the quantum search algorithm allows it to be solved using approximately $\surd(N)$ operations[1][2].

1.3 NP – Complete Problems

In complexity theory, the NP-complete problems are the most difficult problems in NP ("non-deterministic polynomial time") in the sense that they are the ones most likely not to be in P. Formally, a decision problem C is NP-complete if it is complete, it is in NP and it is NP-hard, i.e. every other problem in NP is reducible to it[8].

Some well-known problems that are NP-complete when expressed as decision problems are Hamiltonian Cycle problem, Traveling salesman problem, Subgraph isomorphism problem, Graph coloring problem, Boolean satisfiability problem(SAT) etc.

2. HAMILTONIAN CYCLE PROBLEM

In graph theory, the Hamiltonian cycle problem is a problem of determining whether a Hamiltonian cycle exists in a given graph. The graph may be directed or undirected. The problem of searching for Hamiltonian cycles, or circuits in a given graph is known to be NP-complete, that is, non-determinant in a classical computer, and always polynomial in a massively parallel processor. Ability to solve such problems might benefit many areas, including the layout of integrated circuits. The Hamiltonian cycle problem is a special case of the traveling salesman problem, the exact solution to this problem may be found as a Hamiltonian circuit with minimum total weight[3]. Once all Hamiltonian circuits are identified, it is easy to calculate weight using an ordinary computer to choose the minimum.

2.1 Problem Definition

Let G = (V, A) be a graph in which V = $\{v_1, v_2 \dots v_n\}$ is the set of n vertices, and A is the set of m arcs (v_i, v_j). A Hamiltonian cycle (or circuit) in G is a permutation (s_i) of the vertices such that (v_s, v_{s+1}) belongs to A for i = 1, 2, …, n-1. Also (v_{sn}, v_{s1}) must belong to A to close the circuit.

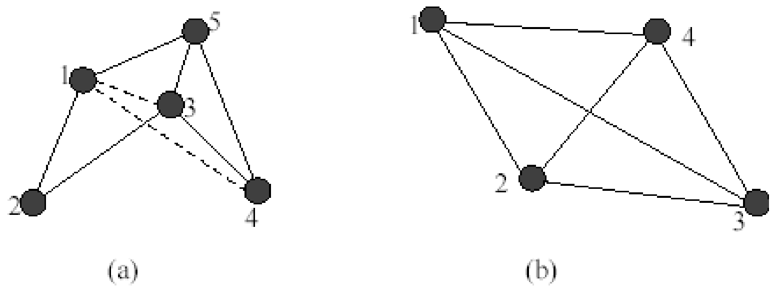

Figure 1. Examples of Hamiltonian circuits

Let us consider the above two circuits which characterizes the presence or absence of Hamiltonian circuits. Figure 1a, is a Hamiltonian circuit 1-2-3-4-5-1. Examples of circuits that are not Hamiltonian are 1-2-3-5-1 and 1-2-3-5-4-5-1 because the first leaves out node 4 while the second visits node 5 twice. Figure 1b is characterized by several Hamiltonian circuits: 1-2-3-4-1; 1-2-4-3-1; 1-3-2-4-1; each can be walked backwards, for example: 1-4-3-2-1.

2.2 Classical Approach

A typical algorithm to solve Hamiltonian circuit classically is to perform a search through all possible orderings of the vertices:

1. Generate each possible orderings ($v_1, v_2, \ldots, v_n$) of vertices
2. For each ordering, check to see whether it is a Hamiltonian cycle for the graph.
   If not, continue checking the orderings.

Since there are $n^n = 2^{nlogn}$ possible orderings of the vertices which must be searched, this algorithm requires $O(p(n)2^{n[\,logn]})$ operations to check for the existence of a Hamiltonian cycle. The polynomial factor p(n) which is predominant due to the implementation of the oracle. This algorithm is deterministic and succeeds with probability [2]. This approach works well for smaller numbers of vertices, but grows exponentially with n, the number of vertices [4].

2.3 Quantum Approach

Nielsen and Chuang began the process of finding Hamiltonian circuits using a quantum computer. On a quantum computer it is possible to estimate the number of solutions much more quickly than is possible on a classical computer by combining the Grover iteration with the phase estimation technique based on quantum Fourier transform [2]. And this is referred to as quantum counting. This allows us to decide whether or not a solution exists, depending on whether the number of solutions is zero, or non-zero.

Let m ≡ [log n], the search space for the algorithm be represented by a string mn qubits, with each block of m qubits being used to store the index to a single vertex with n bits. Therefore, we can write the computational basis state as $|v_1, v2,..,v_n>$, where each $|vi>$ is represented by the approximate string of m qubits, for a total of nm qubits.

The oracle is a unitary operator, *O*, defined by its action on the computational basis:

$|x>|q> \xrightarrow{O} |x>|q \text{ XOR } f(x)>$ where $|x>$ is the index register, the oracle qubit $|q>$ is a single qubit which is flipped if f(x) = 1 and it is unchanged otherwise, XOR denotes modulo addition 2**.**

The oracle for the search algorithm must apply the transformation:

$$O|v_1,v_2,..,v_n> = \begin{cases} |v_1,v_2,..,v_n> & \text{if } v_1,v_2,\ldots,v_n \text{ is not a Hamiltonian cycle} \\ -|v_1,v_2,..,v_n> & \text{if } v_1,v_2,\ldots v_n \text{ is a Hamiltonian cycle} \end{cases}$$

This oracle is easy to design and implement if the description of the graph is known. Applying this oracle, the quantum algorithm require $O(p(n)2^{n[\log n]/2})$ operations to determine whether a Hamiltonian cycle exists. Due to the implementation of the oracle, the polynomial p(n) becomes the overhead which is predominant. But the dominant effect is in determining the resources required for computation which is the exponent in $2^{n[\log n]/2}$. By repeating the algorithm several times (say r) the probability of error can be reduced from 1/6 to $1/6^r$

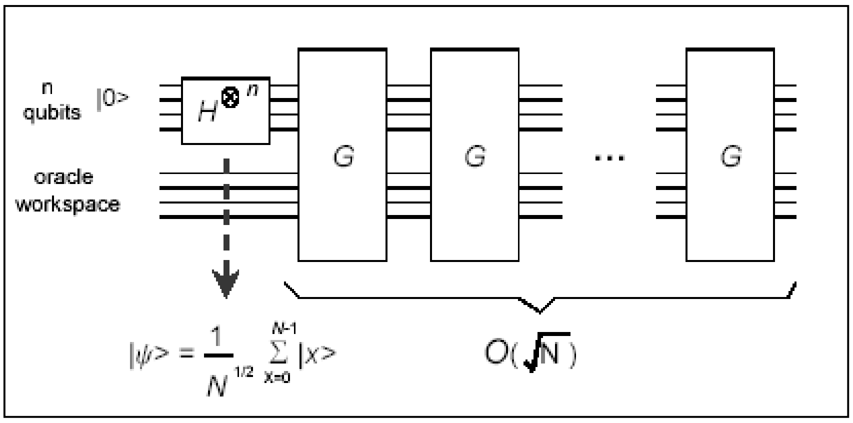

Figure 2. Schematic circuit of Grover search algorithm[2]

2.4 Performance Comparison

The Table 1 given below compares the linear search with quantum search techniques. The comparisons are based on the number of operations required to search the list of N items and the number of operations required to determine the existence of a Hamiltonian in the given circuit. Table 2, gives the performance of classical and quantum search algorithms with respect to the number of vertices & utilization of computational resources.

Table 1. Comparison of classical and quantum search algorithms

Algorithm	Classical search (linear)	Quantum search
Executes on	Classical computer	Quantum computer
Uses	Bits	Qubits
Operations to search N items	O(N)	O(√(N)
Operations to determine the existence of a HC	$O(p(n)2^{n[\log n]})$	$O(p(n)2^{n[\log n]/2})$

Table 2. Performance of classical and quantum search algorithms with respect to the number of vertices & utilization of computational resources

No. of Vertices	Classical search resources required for computation $2^{n[\log n]}$	Quantum search resources required for computation $2^{n[\log n]/2}$
2	1.52	1.23
3	2.69	1.64
4	5.30	2.29
5	11.27	3.36
6	25.43	5.04
7	60.36	7.77
8	149.57	12.23
9	384.85	19.62
10	1024.0	32.0
20	68073969.9	8250.69
100	1.61×10^{60}	1.26×10^{30}

The figure shown below is the graphical representation of the performance analysis of the classical and quantum algorithms in terms of the number of vertices and the amount of computational resources required. It can be seen that the resources required for computing the existence of Hamiltonian remains almost same for both classical and quantum algorithms when the number of vertices happen to be very small. However, as the number of vertices increase tremendously, quantum search takes over the classical in terms of speed.

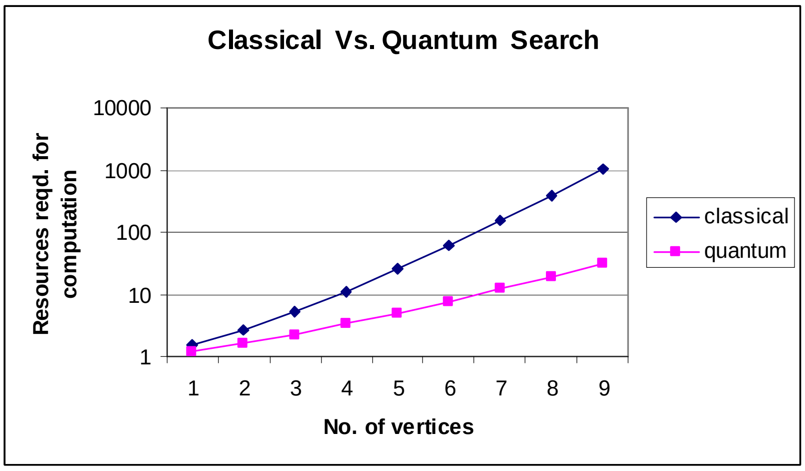

Figure 3. Graphical representation of performance of classical Vs. quantum algorithm

3. CONCLUSION

In this paper, we studied and analyzed undirected graphs and we were able to apply classical search techniques to find out the existence of Hamiltonian cycle in it. Later we applied Grover's quantum search algorithm to find the solution to the problem of Hamiltonian cycle, using the smallest possible number of applications of oracle. Hamiltonian cycle problem is a NP-complete problem, and by performing exhaustive searches over the set of possible solutions, it would result in a considerable speedup over classical solutions.

ACKNOWLEDGEMENTS

We would like to thank all the authors of various papers and publishers for giving us enough information on this emerging area of computing. Our thanks are due to our institute, “The National Institute of Engineering, Mysore” for all the encouragement and support rendered.

REFERENCES

1. Grover L.K, A Fast Quantum mechanical Algorithm for Database Search, In proceedings of the 28th Annual ACM Symposium on the Theory of Computing, pp. 212-219, quant-ph/9605043, (1996).

2. Nielsen M and Chaung I, Quantum Computation and Quantum Information, Cambridge University press, Cambridge, United Kingdom (2000).

3. G. Johnson, A shortcut through time The path to the quantum computer, Vintage Books, (2003).

4. S. Martello, Hamiltonian Circuits in a Directed Graph, ACM Trans. Math. Software, Vol. 9, No. 1, pp. 131-138, (1983).

5. Burger, John Robert, Quantum Algorithm Processors to Reveal Hamiltonian, arXiv:cs/0508116, 08/ (2005).

6. P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Computing 26, (1997).

7. Deutch, D., Quantum Theory, the Church-Turing Principle, and the Universal Quantum Computer, proc. Roy. Soc. Lond. A400, (1985).

8. M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and company, (1995).

9. L.K. Grover, Quantum mechanics helps in searching for a needle in a haystack, Phys. Rev. Lett., 78, pp. 325-328 (1979).

10. Peter W. Shor, Progress in quantum algorithms, MIT, Sept.(2005).