International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 09 | Sep 2020 www.irjet.net p-ISSN: 2395-0072

A SURVEY ON SUBSTANTIAL SUPER COMPUTERS

Mrs. S. Shanthi[1] Mrs. V. Indira[2]

*[*1]Assistant Professor, Department of Computer Science, D.K.M College for Women (Autonomous), Vellore. Tamilnadu, India.*

*[*2]Research Scholar Department of Computer Science, D.K.M College for Women (Autonomous), Vellore. Tamilnadu, India.*

Abstract- Quantum computing is an emerging technology. The clock frequency of current computer processor systems may reach about 40 GHz within the next 10 years. By then, one atom may represent one bit. Electrons under such conditions are no longer described by classical physics, and a new model of the computer may be necessary by that time. The quantum computer is one proposal that may have merit in dealing with the problems presented. Currently, there exist some algorithms utilizing the advantage of quantum computers. For example, Shorts algorithm performs factoring of a large integer in polynomial time, whereas classical factoring algorithms can do it in exponential time. In this paper we briefly survey the current status of quantum computers, quantum computer systems, and quantum simulators.

Key Words: **Classical computers, quantum computers, quantum computer systems, quantum simulators, Shorts algorithm**

I. INTRODUCTION

How much can the performance of a computer beim-proved? According to More law, if the performance keeps improving by means of technological innovations, which has occurred over the last few decades, the number of transistors per chip may be doubled every 18 months. Fur-thermore, processor clock frequency could reach as much as 40 GHz within 10 years [1]. By then, one atom may represent one bit [1]. One of the possible problems may be that, because electrons are not described by classical physics but by quantum mechanics, quantum mechanical phenomenon may cause tunneling to occur on a chip. In such cases, electrons could leak from circuits. Taking into account the quantum mechanical characteristics of the one-atom-per-bit level, quantum computers have been proposed as one way to effectively deal with this predicament. In this way, quantum computers can be used to solve certain computationally intense problems where classical computers. Quire large amounts of processing time. Notwithstanding, further improvements will be necessary to ensure quan-tum computers proper performance in future, but such improvements seem obtainable.

Currently, there exist some algorithms utilizing the advantage of quantum computers. For instance, the polynomial-time algorithm for factoring a large integer with O(n3) time was proposed by Peter Shor [2]. This algorithm performs factoring exponentially faster than classical computers. This algorithm could factor a 512-bit product in about 3.5 hours with 1 GHz clock rate [3], whereas the number bled sieve could factor the same product in 8400 MIPS years [4]. (One MIPS year is the number of instructions that a processor can execute in a year, at the rate of millions of instructions per second.) Another famous quantum algorithm is a database search algorithm proposed by Lov Grover that will and a single item from an unsorted list of N elements with O(N) time [5].

In this paper we briefly survey quantum computers. First, the main characteristics of quantum computers, superposition states, and interference are introduced. Then, current approaches to quantum computers are reviewed. Next, research on quantum computer simulators is introduced. We conclude with a few remarks.

2. Quantum Computer Systems 2.1 Superposition State

In classical computers, electrical signals such as voltages represent the 0 and 1 states as one-bit information. Two bits indicate four states 00, 01, 10, and 11, and n bits can represent 2^n states. In the quantum computer, a quantum bit called qubit which is a two-state system, represents the one-bit information. For instance, instead of an electrical signal in classical computers, an electron can be used as a qubit. The spin-up and spin-down of an electron represent two states: 0 and 1, respectively. A photon can also be used as a qubit, and the horizontal and vertical polarization of a photon can be used to represent both states. Using qubits, quantum computers can perform arithmetic and logical operations as does a classical computer. The important difference, however, is that one qubit can also represent the superposition of 0 and 1 states. When we represent 0 and 1 states as state vectors |0 and |1 respectively, such a superposition state is expressed as a linear

This bizarre characteristic in quantum computers makes parallel computation possible in the real sense of the term. Because each qubit represents two states at the same time, two qubits can represent four states simultaneously. For instance, when we use two qubits that

are the super-position of 0 and 1 states as an input for an operation, we can get the result of four operations for four inputs with just one computational step, as compared to the four operations needed by the classical computer. Likewise, when using n qubits, we can make a superposition of 2^n states as an input and process the input in just one step to solve a problem for which a classical computer requires 2^n steps. In this light, a quantum computer can process n inputs with only one computational step after taking the superposition state of n inputs.

However, there is a crucial problem to solve before we can use this extremely valuable characteristic of quantum computers. From the input of one superposition state representing four states and processing in one step we get the superposition of four results. When we measure the output qubits, the quantum mechanical superposition collapses and each qubit will be observed as either 0 or 1 because a qubit is a two-state system. Consequently, we only get one of the four possible results: 00, 01, 10, or 11 (for n = 2) with the same probability. Accordingly, the superposition of qubits is governed by probability, and the measurement is necessary to determine which one of the possible states is represented. This difficulty arises from using the quantum mechanical superposition. If, however, we can increase the probability of getting the expected result by devising an algorithm, we may take advantage of the quantum mechanical superposition feature. In this way, as discussed above, we can harness the power of quantum computers to solve a problem that takes an excessive amount of computational time and energy for certain problem classes on classical computers.

2.2 Interference

In this subsection, we give a simple example that illustrates the difference between classical and quantum computation, and the importance of interference-of-states in quantum computation.

Clearly, any classical computer can be simulated by a Turing machine, a mathematical model of a general computer. Before we discuss the quantum Turing machine (QTM), we introduce a computation tree using a classical probabilistic Turing machine (PTM) [6]. Fig. 1 shows an example of a state transition diagram for the PTM, and Fig. 2 derives the PTM as a computation tree. In the tree, each vertex shows a machine state and each edge shows the probability of transition occurrence

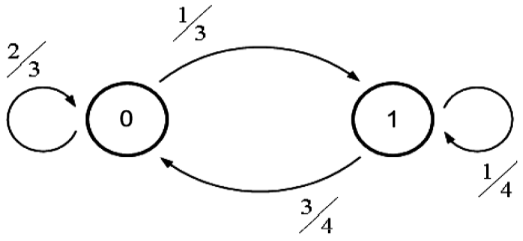

Figure 1. A state transition diagram of PTM

Also, each level of the tree represents a computation step and the trees root represents the starting state. We can compute a probability of transition 0 1 after two computational steps, by summing the probabilities of the two possible paths from the root to state 1 as follows:

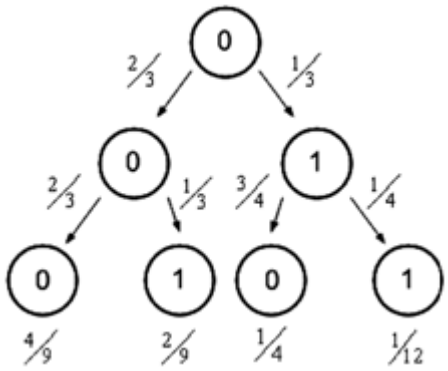

Figure 2. A computation tree of PTM

$$P(0\ \ 1) = \frac{2}{3}\ \frac{1}{3} + \frac{1}{3}\ \frac{1}{4} = \frac{2}{9} + \frac{1}{12} = \frac{11}{36}$$

Similarly:

$$P(0\ \ 0) = \frac{2}{3}\ \frac{2}{3} + \frac{1}{3}\ \frac{3}{4} = \frac{4}{9} + \frac{3}{12} = \frac{25}{36}$$

We can interpret this result in the following way. In two steps, starting from state 0 the PTM will occupy state 1 with probability 11/36 and state 0 with probability 25/36. Similar to PTM, we describe a computation of QTM using the computation tree shown in Fig. 3. Each edge of the tree in QTM represents a probability amplitude, whereas in the PTM each edge represents a transition probability. Only one state in the same level of the PTM tree occurs at a time, but all states in the same level of the QTM tree occur simultaneously! For this example, the probability of 0 1 from the root after one computational step is:

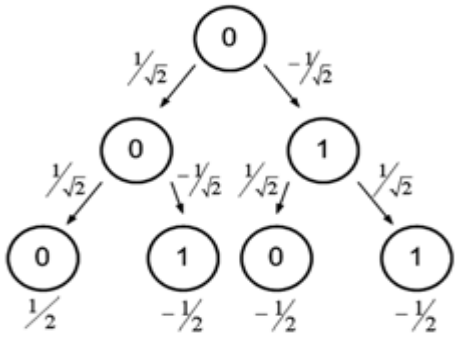

Fig: QTMS Starts from State0

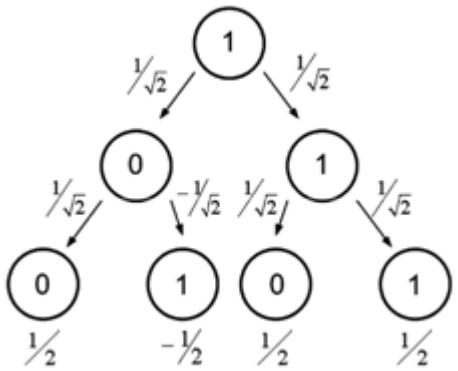

Fig: QTMS Starts from State0

Let us compute the probability of transition 0 1 after two steps. First, we need to Þnd the probability mplitudes

$$\Psi(0\ \ 0\ \ 1) = \frac{1}{2}\ \ \frac{1}{2} = \frac{1}{2}$$

$$\Psi(0\ \ 1\ \ 1) = \frac{1}{2}\ \ \frac{1}{2} = \frac{1}{2}$$

Where

This is a remarkable result. After one computational step, the probabilities 0 1 and 0 0 were both 1/2. But after two computational steps from the same root the probability 0 1 is 1 and probability 0 0 is 0. This result occurs because the probability amplitudes can have negative values. We interpret this result as due to the states of the QTM interfering with each other. In short, the case Ò0 1 after two steps had a constructive interference [((1/2)) + ((1/2)) = 1] and the case Ò0 0 after two steps had a destructive interference [(1/2) + ((1/2)) = 0].

In the previous subsection, we mentioned that the result of a computation involving the superposition of n input states is a superposition of n-output states. For example, if we need to perform factorizing of an n-digit binary number into two prime factors, we must test 2^{n-1} numbers with Eratosthenes sieve as the worst-case scenario. Therefore, we must make a superposition of 2^{n-1} integers as input giving the result from factoring as the superposition of 2^{n-1} outputs.

If we can design an operation such that a constructive interference occurs at desired outputs (e.g., prime factors) of the superposition of 2^{n-1} outputs and a destructive interference occurs at unnecessary outputs, we can Þnd prime factors with only one computational step as compared to the classical computer, which takes 2^{n-1} steps. This is an immense improvement in computation time

Shorts algorithm performs factoring of large integers, though it is not just a single-step operation as described. The algorithm consists of both quantum and classical processing. The quantum processing part utilizes quantum interference and the superposition state to and the period r of the function $f_{x,n}(a) = x^a \bmod n$ where n is an integer to be factored and x is an integer chosen at random that is coprime to n (i.e., gcd(x, n) = 1). The classical part makes use of a result from classical number theory to Þnd a factor of n by using x and r from the quantum part.

II. Current Approaches to Quantum Computers

In this section we consider how such a quantum computer can be built. There are Þve experimental requirements for building a quantum computer [8, 9]. The requirement is the ability to represent quantum information robustly.

Because a qubit is a simple two-level system, a physical qubit system will have a Þnite set of accessible states. Some examples are the spin states of a spin 1/2 particle, the ground states and Þrst excited states of an atom, and the vertical and horizontal polarization of a single photon. Second, a quantum computer requires the ability to set a fiducial initial state. This is a significant problem for most physical quantum systems because of the imperfect isolation from their environment and the difficulty of producing desired input states with high. Third, a quantum computer requires long decoherence times, much longer than the gate operation time. Decoherence is the coupling between the qubit and its environment, which results in a loss of the quantum phase coherence. After decoherence, the quantum mechanical property associated with coherence (e.g., superposition, entanglement) can no longer be observed. The fourth requirement is the capability of measuring output results from specific qubits. The outcome from a quantum algorithm is, in general, a quantum superposition. Therefore, it is necessary to read out a result from the quantum state using the classical system with highbreifly. The fifth requirement concerns the ability to construct a universal set of quantum gates. Similar to a classical computer, a quantum computer has universal gates, which implement any legitimate quantum computation. It proved that just two-qubit gates at a time are adequate to build a general quantum circuit

Several implementations for a quantum computer have been proposed. One of the well-researched implementations is a nuclear magnetic resonance (NMR) based quantum computer. This computer uses a vial of a liquid filled with sample molecules as qubits. In this way, this experimental quantum computer solves a problem by controlling nuclear spins using NMR techniques and retrieves the results observing the ensemble average of some property of the nuclear spins in the vial. A seven-qubit NMR-based quantum computer has been built, and the computer can perform Shorts algorithm finding factors of the number 15 [11]. This is currently the most advanced quantum computer

A superconducting quantum computer uses the Josephson-junctions in superconducting circuits as qubits Charge or energy levels in a junction represent information of qubits. A controlled-NOT gate operation on the charged qubits was demonstrated, but the phase evolution during the gate operation has not yet been examined [21]. An implementation of the real quantum controlled-NOT gates is the next challenge in the realization of universal logic gates.

Although each proposed quantum computer has difficulties in its realization, a common critical problem is that real quantum memory registers incur errors caused by environmental coupling (e.g., cosmic radiation, spontaneous emission, and decoherence). As it is extremely difficult to isolate quantum registers perfectly from their environment, a real quantum computer must be designed considering the effect of errors on the state of the quantum registers.

To protect quantum states against the effects of noise, several quantum error-correcting (QEC) schemes have been proposed [22Ð25]. QEC codes could be developed based upon principles similar to a classical error-correcting code. However, we need to circumvent the following three difficulties to design a QEC code [8]. First, we cannot pro-duce a repetition code (e.g., logical 0 and 1 is encoded as Ò0000Ó and Ò1110Ó respectively) by duplicating the quantum state several times because the no-cloning theorem [26] states that replication of an arbitrary quantum state is not possible. Second, unlike a classical bit, inspecting the state to assess its correctness can destroy a qubit. Third, because the state of qubit depends on certain continuous parameters (e.g., a rotation angle θ), quantum errors are continuous. Consequently, infinitive precision is required to determine which error occurred to correct them.

By implementing the QEC codes on a quantum circuit, we can reduce the effect of noise on quantum registers and transmissions. However, it is not sufficient for quantum computation because in practice gate operations (e.g., encoding, decoding, and error correction) on the quantum circuit are themselves prone to errors. Moreover, these errors are propagated and accumulated continuously until the computation is completed.

CONCLUSION

In this paper we have reviewed the principles, algorithms, and hardware considerations for quantum computing. Several research groups are investigating qubits and quantumlogic circuitry using different resources (i.e., atom, ion, electron, and photon, among others). The realization of a practical quantum computer is expected before we encounter the limit of Moores law with respect to improvements that may be possible using the classical computer model. A current realizable quantum computer is based on seven-bit NMR, which can factor 15. Further research is needed, for example, via simulation, on quantum computers using classical computers. Such a simulator must be able to handle quantum computers that operate on a practically large number of qubits. To this end, we need to employ large-scale parallel processing methods to acquire more meaningful results within a practical time frame. By applying the methods/concepts of classical computers such as hardware abstraction to quantum computers, the research progress may be accelerated. For example, some groups proposed quantum programming languages that allow us to think of quantum computer operations in an abstract manner as we do with a classical computer. Efforts at realization for quantum computers have just begun. Undoubtedly, we need more intensive research in a physical realization of components of quantum computers Computer scientists/engineers will need to consider the various architectural solutions for quantum computers as well as the various new (practical) quantum algorithms to advance the state of the art for quantum computers

REFERENCES

[1] C.P. Williams & S.H. Clearwater, **Exploration in quantum computing** (New York: Springer-Verlag, 1997).

[2] P.W. Shor, Algorithm for quantum computation: Discrete logarithm and factoring, **Proc. 35th IEEE Annual Symp. On Foundations of Computer Science**, Santa Fe, NM, November1994, 24Ð134.

[3] M. Oskin, F.T. Chong, & I. Chuang, A practical architecture for reliable quantum computers, **IEEE Computer**, January 2002, 79Ð87.

[4] B. Preneel (Ed.), Factorization of a 512-bit RSA modules, **Lec-ture Notes in Computer Science**, Vol. 1807 (Berlin: Springer-Verlag, 2000).

[5] L.K. Grover, A fast quantum mechanical algorithm for database search, **Proc. STOC**, Philadelphia, 1996, 212Ð219.

[6] D.R. Simon, on the power of quantum computation, **Proc. 35thAnnual Symp. On Foundations of Computer Science**, Sante Fe,NM, 1994, 116Ð123.

[7] T. Nishino, **Introduction to quantum computer** (Tokyo: Tokyo Denki University Press, 1997) (in Japanese).

[8] M.A. Nielsen & I.L. Chuang, **Quantum computation and quantum information** (Cambridge: Cambridge University Press, 2000).

[9] D.P. DiVincenzo, The physical implementation of quantum computation, **Fortschritte der Physik**, **48**(9Ð11), 2000, 771Ð783.

[10] D.P. DiVincenzo, Two-bit gates are universal for quantum computation, **Physical Review A**, **51**, 1995, 1015Ð1022.

[11] IBM Research News, **IBM's test-tube quantum computermakes history: First demonstration of Shor's historic factor-ing algorithm**, http://www.research.ibm.com/resources/news/20011219_quantum.shtml.

[12] J.I. Cirac& P. Zoller, Quantum computations with cold trapped Ions, **Physical Review Letters**, **74**, 1995, 4091.

[13] C. Monroe, D.M. Meekhof, B.E. King, W.M. Itano, & D.J. Wineland, Demonstration of a fundamental quantum logic gate, **Physical Review Letters**, **75**, 1995, 4714.

[14] D. Kielpinski, C. Monroe, & D.J. Wineland, Architecture for a large-scale ion-trap quantum computer, **Nature**, **417**, 2002, 709Ð711.

[15] Q.A. Turchette, C.J. Hood, W. Lange, H. Mabuchi, & H.J. Kimble, Measurement of conditional phase shifts for quantum logic, **Physical Review Letters**, **75**, 1995, 4710Ð4713.

[16] E. Knill, R. Laßamme, & G. J. Milburn, A scheme for efficient quantum computation with linear optics, **Nature**, **409**, 2001, 46Ð52.

[17] T.C. Ralph, A.G. White, & G.J. Milburn, Simple scheme for efficient linear optics quantum gates, **Physical Review A**, **65**, 2002, 012314-1Ð012314-6.

[18] D. Loss & D.P. DiVincenzo, Quantum computation with quantum dots, **Physical Review A**, **57**, 1998, 120Ð126.

[19] M.S. Sherwin, A. Imamoglu, & T. Montroy, Quantum com-putation with quantum dots and terahertz cavity quantum electrodynamics, **Physical Review A**, **60**, 1999, 3508Ð3514.

[20] Y. Makhlin, G. Sch¬on, &A. Shnirman, Quantum-state engineering with Josephson-junction devices, **Reviews of ModernPhysics**, **73**, 2001, 357Ð400.