Cogn Comput (2010) 2:63–67
DOI 10.1007/s12559-010-9032-9

Quasi-Quantum Computing in the Brain?

Pentti O. A. Haikonen

Published online: 23 February 2010

Abstract Quantum computing has been seen as a potentially powerful computing method, and consequently some researchers have argued that the brain might somehow utilize quantum computing processes. In the laboratory, quantum computing calls for exotic conditions, and it has been argued that the brain cannot provide these. Here, a novel computing method, quasi-quantum computing, is presented. This method utilizes the main principles of quantum computing: superposition, entanglement and collapse, but in this case, computing is not based on quantum processes; instead, it is realized by utilizing conventional electronic devices in rather unconventional ways. As an example of the potential of this approach, the reverse computing of mathematical functions is considered, and an experimental test device is reported. Some aspects of this approach may be used by the brain as no exotic conditions are required. However, further research would be required.

Keywords Computational networks · Quantum computing · Reverse computing · Cognitive computing · Factorization

Introduction

Human cognition is still somewhat mysterious, and the actual brain process that manifests itself as thinking is not known in sufficient details. Artificial neural networks have been developed for models of the brain. However, this work is not without problems, and further research is necessary. The biological neural networks of the brain are rather slow, and it is not obvious how they could compute in real time everything that is necessary. For instance, small birds have small brains, yet they manage to control their flight through thorny bushes so that they will not hurt themselves. Consequently, some researchers have argued that the computational power of the brain is not due to the rather simple interaction between neurons; instead, it would ultimately rely on quantum processes [1]. According to the quantum mind theory, conscious problem solving involves the evolution of the quantum wave function into a superposition of a number of states. The collapse of these states into a set of definite states would give the solution to the problem [2]. On the other hand, it has been argued that the brain cannot support the required quantum coherence due to its rather high working temperature. It has also been argued that the conditions for the Schrödinger equation are not met in the brain because synapses and neurons do not conserve energy [3].

Quantum entanglement and superposition may enable massively parallel quantum computers [4]. According to the quantum theory, an observable property of two entangled particles is indeterminate for both particles until a measurement is made. The eventual measurement will show that the measurement results for both particles will be correlated no matter how far the particles are from each other. Until this measurement, the quantum states of these particles are in a superposition state where 0 and 1 coexist. This property is utilized in the quantum computer where the computational unit, the quantum bit (qubit), can be in the superposition state. An example of the computations where quantum computers may excel is the factorization of large integers. The product of two large integers can be easily computed, but if the product is given, the finding of

P. O. A. Haikonen (✉)
Department of Philosophy, University of Illinois at Springfield, One University Plaza, Springfield, IL 62703, USA
e-mail: pentti.haikonen@pp.inet.fi

the factors can be extremely time-consuming with a conventional computer and any of the algorithms known today. However, practical quantum computers are notoriously difficult to build. Quantum superposition states are very fragile and demand exotic equipment and possibly very low working temperatures.

The author presents here another approach to computational networks, one that utilizes similar principles as quantum computing, namely superposition and entanglement. However, this approach does not rely on quantum processes in the quantum computing sense and is executable by standard electronic components at the normal temperature range of those components. It is understood that real qubits cannot be realized with non-quantum standard electronic components; however, there are ways of emulating some of the features of quantum computing. Because of the superficial similarities between real quantum computing and the presented method, the author calls this method "quasi-quantum computing". In the following, a specific application of this method, which the author calls bi-directional Boolean logic, is used to illustrate the principles of this approach.

Quasi-Quantum Computing with Bi-Directional Boolean Logic

Standard Boolean logic can be used to execute binary calculations such as the addition and multiplication of binary numbers. When input numbers are inserted into the computational Boolean network, signal paths emerge and converge at the correct output. The point is: at that stage, the signal paths are logically consistent in both directions, forward and reverse. This fact would seem to allow the computation of the function in the reverse order, from the result backwards to the initial values. However, there are two practical problems here. First, the existing electronic Boolean logic circuits (Integrated Circuits) are not reversible; they do not translate their output values back into input values. Secondly, the reversing of most Boolean logic truth tables leads to undetermined states where the logical one and zero would seem to be equally likely. This state resembles the quantum superposition state. The Boolean AND function illustrates the problem of reversibility. A two-input AND is considered here with the inputs A and B and the output C. The output C has the logical value of 1 only when both inputs A and B are 1. Whenever $C = 0$, A and B may have the following value pairs: {0,0}, {0,1}, {1,0}. Thus, whenever $C = 0$, the corresponding values of A and B cannot be determined, they may have the values of 0 and 1. When this ambiguity is marked ∅, the following reversed truth table results:

C	A	B
0	∅	∅
1	1	1

The ambiguity state ∅ can be understood as a kind of superposition of the 1 and 0 states. (This is a logical superposition, not the quantum one; the term "superposition" is used here because it describes the coincident possibility of logical zero and one.) In an actual hardware realization, the A and B circuit terminals would have to be able to exhibit an undetermined logical state in addition to the logical zero and one. This ambiguity state cannot be resolved without additional information. However, we can see that in this superposition (when $C = 0$), A and B are entangled:

if A = 1 then B = 0
if B = 1 then A = 0.

By the entanglement of A and B, it is meant here that the logical states of A and B are not independent of each other but have a two-way dependence. As such it is functionally rather similar to the entanglement that is used by traditional quantum computing. This function is different from the standard logic element function, which works only in one direction.

Thus, a reverse AND circuit would accept C as the input and have A and B terminals that could sustain superimposed logical states. Additionally, the A and B terminals would be logically connected with the entanglement rule. This entanglement, in order to be effective, would necessitate a dual nature for the A and B terminals; they would have to act as inputs or outputs as needed. Here an AND circuit, which operates normally in forward direction and additionally realizes the superposition and entanglement as described before, is called the bi-directional AND (biAND). The conventional AND and the corresponding bi-directional AND are depicted in the Fig. 1.

The biAND function can be realized by standard electronic components. Fig. 2 gives one possible realization that utilizes the very high input impedance (Gigaohms) of high-speed C-MOS logic components.

In Fig. 2, IC1 and IC2 are tri-state logic buffers of the type 74HC126. The AND1 circuit is of the type 74HC08. All diodes are 1N4148.

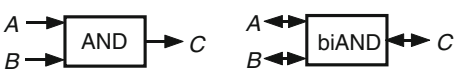

Fig. 1 Two-input AND and the corresponding bi-directional AND (biAND)

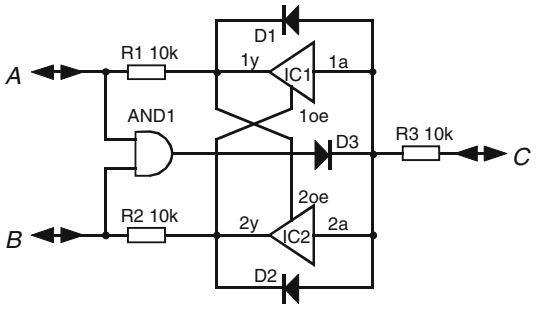

Fig. 2 A biAND circuit realization

The biAND circuit operates in the forward direction (A, $B \rightarrow C$) normally; the AND1 performs the standard AND operation. The result is passed to the terminal C via the diode D3. Thus, whenever both A and B have the logical value one (+5 V), the terminal C will also have the same value.

In the reverse direction ($C \rightarrow A$, B), the circuit operates as follows. Whenever C has the value zero, the diodes D1 and D2 do not conduct, and the outputs of the buffers IC1 and IC2 stay at the high impedance state. If now one of the terminals, say A, is forced to logical one, then the output enable *2oe* will go high and the IC2 will pass the value at the terminal C (0 V) to the terminal B. Thus, the entanglement rule has been implemented. Whenever C has the logical value one (+ 5 V), the diodes D1 and D2 conduct and logical one is passed to the terminals A and B. The resistors R1 and R2 act as current limiters in case of high–low state contests. This biAND realization is a latching circuit that must be reset by powering it down.

If the terminal C of a biAND circuit is set to zero, the terminals A and B will have ones and zeros superimposed; they will not represent and transmit logical zeros or ones but are ready to receive external information. This superposition will collapse if one of the terminals A or B is forced to logical one. This kind of collapse can take place in networks consisting of interconnected logic circuits. A simple example of this operation is given in Fig. 3.

In Fig. 3, the terminals *B1* and *A2* of the biAND1 and biAND2 are tied together. When the terminal *C1* of the biAND1 is set to one, *A1* and *B1* will have the logical value one. As the terminal *B1* is connected to the terminal *A2*, the

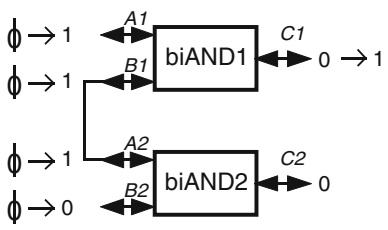

Fig. 3 Superposition and entanglement in a network

superposition state at *A2* will be destroyed, and consequently the entanglement rule forces *B2* to the value zero, which is then transmitted to any elements that are connected to that terminal.

A bi-directional Boolean NOT function (inverter) can be devised in a similar style. Every other Boolean function can be realized by combinations of NOT and AND functions, and this is applicable to the bi-directional logic functions, too. However, more complex reverse logic circuits can also be designed directly from their truth tables. The general logic element with combined input/output terminals and logical entanglement rules is described in the U.S. Patent No. 7,161,385 [5].

A bi-directional logic network consists of bi-directional logic elements that are interconnected to solve the requested function in the forward direction. When the function is solved, the network will settle in logical equilibrium where the logical states of the nodes are not contested. The network will then solve the requested function also in the reverse direction if certain conditions are met.

An Application to Reverse Multiplication

Multiplication is one mathematical operation that is easy to execute in the forward direction but tedious to execute in the reverse direction when the factors of a number are to be found. This property is utilized in several data encrypting schemes; in order to crack the encrypting, one would have to factorize a large number that is the product of two large primes.

Factorization is one possible application of bi-directional logic. In this application, the factorization apparatus is designed as a reverse binary multiplier; the logic elements that would constitute the normal binary multiplier are replaced with the corresponding bi-directional ones. Thereafter, the original output of the circuit, the product, will now be the input, and the original inputs will appear as the resulting factors. The number to be factorized is set as the output, and the factors can be read from the original inputs. The execution speed for the factorization operation will be about the same as for the forward multiplication.

As a practical test of the concept, the author has built along these principles a binary reverse multiplier, which accepts a 6-bit binary number as an input and factorizes it into two 3-bit factors, see Fig. 4. The factorization can lead at least to two possibilities (e.g. $15 = 3 \times 5$ or 5×3). Therefore, at least two different logical paths exist. This symmetry of choice must be broken to make the network settle. This can be achieved by weak DC bias signals that can be overruled by the net should they eventually contest the emerging logical equilibrium. Also low-level noise may be used. The bias signals alone will not cause any output

i.e. $0 = 0 \times 0$. Bias signals can be inserted into the bias points indicated in fig. 4.

The experimental binary reverse multiplier factorizes correctly the following numbers with one set of bias signals instantly at power-on: $0 = 0 \times 0, 8 = 2 \times 4, 12 = 3 \times 4, 15 = 3 \times 5, \quad 16 = 4 \times 4, \quad 20 = 5 \times 4, \quad 24 = 6 \times 4, 28 = 7 \times 4, 42 = 7 \times 6, 49 = 7 \times 7$. Prime numbers such as 2, 3, 5, 7, 11, 13, 17, 19, 23, 29 and 31 cannot be factorized. This computer cannot factorize numbers where one of the factors is larger than three bits or decimal 7 ($22 = 2 \times 11$, $26 = 2 \times 13, 33 = 3 \times 11, 32 = 4 \times 8, 40 = 5 \times 8$ etc.) due to its capacity limit.

It will be seen that a given set of bias signals will not necessarily lead to correct results for every input product value. Therefore, the outcome should be checked by multiplying the produced [*a2 a1 a0*] and [*b2 b1 b0*] binary numbers. Thus, the following procedure could be used: (1) apply a set of bias signals (2) execute reverse calculation (3) check the result by multiplication (4) if not correct apply different bias (5) execute reverse calculation anew (6) check and repeat if necessary.

A number may be a product of different factors, like $12 = 2 \times 6$ or 3×4. Certain bias signals will give one possibility, and other bias signals will reveal other possibilities. An integer (P) that is the product of two primes (A, B) leads to the possibilities $1 \times P$, $P \times 1$, A x B and B x A only and is in that sense easier to factorize by this method.

Figure 5 depicts the experimental quasi-quantum computer. The input binary number to be factorized is set by the six switches on the left. The factors appear on the right as two binary numbers. In this photograph, the input number to be factorized is set to 011000 (decimal 24), and the factors are 110 (decimal 6) and 100 (decimal 4). The operation of this computer is not statistical; with given bias values, the computer outputs always and instantly the same result for a given number to be factorized. (The quasi-quantum computer can be simulated by a circuit simulation program. However, the simulation is usually very slow and will easily stop due to convergence problems.)

This experimental computer is not a full realization of the general principle as its modules do not work in the forward direction; full realization should give additional computing power as this would utilize all available information. It remains to be verified if larger reverse computers could be assembled along the principles presented here.

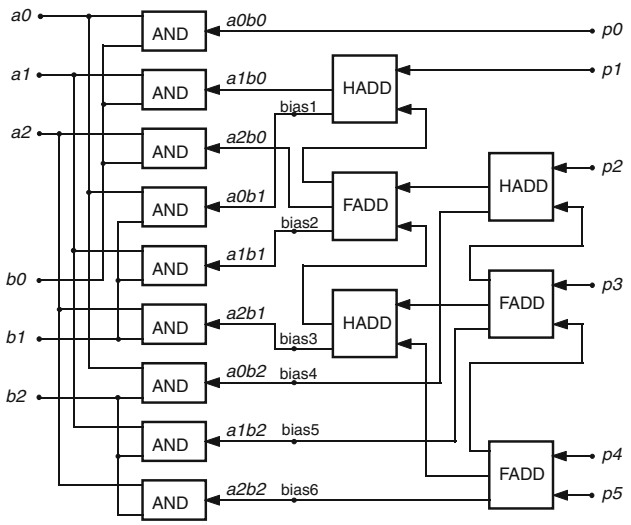

Fig. 4 The block diagram of the binary reverse multiplier (FADD = reverse full adder, HADD = reverse half adder)

Conclusions

Quasi-quantum computing is computing with the non-quantum equivalents of the quantum superposition and entanglement. Quasi-quantum computing is realized with networks of modules that communicate with each other via combined input/output terminals that allow the undecided state; this state accommodates both logical states until the network "collapses" to a logical equilibrium state. Quasi-quantum computing utilizes logical entanglement and superposition that can be realized by conventional electronic components.

Quasi-quantum computing principles allow the creation of bi-directional logical circuits that compute logical functions in forward and reverse directions. The combination of these bi-directional logical circuits should allow the reverse computation of mathematical functions; as a practical test of the concept, a simple reverse multiplier has been built.

The brain utilizes networks of neurons that communicate with each other via electric pulses and chemical

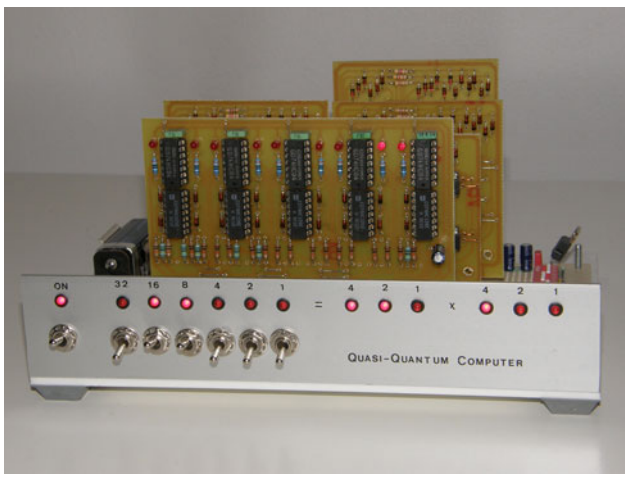

Fig. 5 The experimental quasi-quantum computer

means. It has been speculated that in the brain, the quantum superposition, entanglement and collapse could play a role as these might improve the computing power of the brain. On the other hand, it has been argued that the brain cannot support the required quantum coherence. The principles of quasi-quantum computing might still be applicable to the brain as these do not call for any exotic physical conditions for their operation. The communication between neurons may rely, in one way or other, on kinds of superposition states that collapse to final states aided by learned entanglement rules. At the moment, this hypothesis remains to be verified by further research. It also remains to be seen if this quasi-quantum approach will have any value for practical applications and for the theory of computing, or if this approach will merely remain an intellectual curiosity.

For the sake of clarity, it should be noted that this paper does not seek to criticize or diminish the value of the research on "traditional" quantum computing.

Acknowledgment The author wishes to thank Mr. Dylan Drummond for his kind suggestions about the text.

References

1. Penrose R. The emperor's new mind. New York: Vintage; 1989.
2. Nunn C, Clarke C, Blott B. Collapse of a quantum field may affect brain function. J Conscious Stud. 1994;1:127–39.
3. Scott A. On quantum theories of the mind. J Conscious Stud. 1996;5–6:484–91.
4. Brown J. The quest for the quantum computer. New York: Simon & Schuster; 2001.
5. Haikonen POA. U.S. Patent No. 7,161,385. Circuit elements and parallel computational networks with logically entangled terminals. Granted 01/09/2007.