Probabilistic Quantum Memories

C. A. Trugenberger*
InfoCodex, chemin du Petit-Saconnex 28, CH-1209 Genève, Switzerland
(Received 26 December 2000; published 18 July 2001)

Typical address-oriented computer memories cannot recognize incomplete or noisy information. Associative (content-addressable) memories solve this problem but suffer from severe capacity shortages. I propose a model of a quantum memory that solves both problems. The storage capacity is exponential in the number of qbits and thus optimal. The retrieval mechanism for incomplete or noisy inputs is probabilistic, with postselection of the measurement result. The output is determined by a probability distribution on the memory which is peaked around the stored patterns closest in Hamming distance to the input.

DOI: 10.1103/PhysRevLett.87.067901 PACS numbers: 03.67.Lx, 07.05.Mh

Quantum computation [1] is normally associated with new complexity classes which are inaccessible (in polynomial time) to classical Turing machines. In other words, quantum algorithms [2] can drastically speed up the solution of tasks with respect to their classical counterparts, the paramount examples being Shor's factoring algorithm [3] and Grover's search algorithm [4].

There is, however, another aspect of quantum computation which represents a big improvement upon its classical counterpart. In traditional computers the storage of information requires setting up a lookup table (RAM). The main disadvantage of this address-oriented memory system lies in its rigidity. Retrieval of information requires a precise knowledge of the memory address and, therefore, incomplete or noisy inputs are not permitted.

In order to address this shortcoming, models of associative (or content-addressable) memories [5] were introduced. Here, recall of information is possible on the basis of partial knowledge of their content, without knowing the storage location. These are examples of collective computation on neural networks [5], the best known example being the Hopfield model [6] and its generalization to a bidirectional associative memory [7].

While these models solve the problem of recalling incomplete or noisy inputs, they suffer from a severe capacity shortage. Because of the phenomenon of crosstalk, which is essentially a manifestation of the spin glass transition [8] in the corresponding spin systems, the maximum number of binary patterns that can be stored in a Hopfield network of n neurons is $p_{\max} \simeq 0.14n$ [5]. While various possible improvements can be introduced [5], the maximum number of patterns remains linear in the number of neurons, $p_{\max} = O(n)$.

In this Letter I show that quantum mechanical entanglement provides a natural mechanism for both improving dramatically the storage capacity of associative memories and retrieving noisy or incomplete information. Indeed, the number of binary patterns that can be stored in such a quantum memory is exponential in the number n of qbits, $p_{\max} = 2^n$; i.e., it is optimal in the sense that all binary patterns that can be formed with n bits can be stored. The retrieval mechanism is probabilistic, with postselection of the measurement result. This means that one has to repeat the retrieval algorithm until a threshold T is reached or the measurement of a control qbit yields a given result. In the former case the input is not recognized. In the latter case, instead, the output is determined itself by a probability distribution on the memory which is peaked around the stored patterns closest (in Hamming distance) to the input. The efficiency of this information retrieval mechanism depends on the distribution of the stored patterns. Recognition efficiency is best when the number of stored patterns is very large while identification efficiency is best for isolated patterns which are very different from all other ones, both very intuitive features.

Let me start by describing the elementary quantum gates [2] that I use in the rest of the Letter. First of all there are the single-qbit gates NOT, represented by the first Pauli matrix σ_1, and H (Hadamard), with the matrix representation

$$\mathrm{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1}$$

Then, I use extensively the two-qbit XOR (exclusive OR) gate, which performs a NOT on the second qbit if and only if the first one is in state $|1\rangle$. In matrix notation this gate is represented as $\mathrm{XOR} = \mathrm{diag}(1, \sigma_1)$, where 1 denotes a two-dimensional identity matrix and σ_1 acts on the components $|01\rangle$ and $|11\rangle$ of the Hilbert space. The 2XOR, or Toffoli gate [9], is the three qbit generalization of the XOR gate: it performs a NOT on the third qbit if and only if the first two are both in state $|1\rangle$. In matrix notation it is given by $2\mathrm{XOR} = \mathrm{diag}(1, 1, \sigma_1)$. In the storage algorithm I make use also of the nXOR generalization of these gates, in which there are n control qbits. This gate is also used in the subroutines implementing the oracles underlying Grover's algorithm [2] and can be realized using unitary maps affecting only a few qbits at a time [9], which makes it feasible. All these are standard gates. In addition to them I introduce the two-qbit controlled gates

$$CS^i = |0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes S^i,$$
$$S^i = \begin{pmatrix} \sqrt{\frac{i-1}{i}} & \frac{1}{\sqrt{i}} \\ \frac{-1}{\sqrt{i}} & \sqrt{\frac{i-1}{i}} \end{pmatrix}, \tag{2}$$

for $i = 1, \ldots, p$. These have the matrix notation $CS^i = \mathrm{diag}(1, S^i)$. For all these gates I indicate by subscripts the qbits on which they are applied, the control qbits coming always first.

Given p binary patterns p_i of length n, it is not difficult to imagine how a quantum memory can store them. Indeed, such a memory is naturally provided by the following superposition of n entangled qbits:

$$|M\rangle = \frac{1}{\sqrt{p}} \sum_{i=1}^{p} |p^i\rangle. \tag{3}$$

The only real question is how to generate this state unitarily from a simple initial state of n qbits. To this end one can use the algorithm proposed in [10]. Here, however, I propose a simplified version.

In constructing $|M\rangle$ I use three registers: a first register p of n qbits in which I subsequently feed the patterns p^i to be stored, a utility register u of two qbits prepared in state $|01\rangle$, and another register m of n qbits to hold the memory. This latter is initially prepared in state $|0_1, \ldots, 0_n\rangle$. The full initial quantum state is thus

$$|\psi_0^1\rangle = |p_1^1, \ldots, p_n^1; 01; 0_1, \ldots, 0_n\rangle. \tag{4}$$

The idea of the storage algorithm is to separate this state into two terms, one corresponding to the already stored patterns and another ready to process a new pattern. These two parts are distinguished by the state of the second utility qbit u_2: $|0\rangle$ for the stored patterns and $|1\rangle$ for the processing term.

For each pattern p^i to be stored one has to perform the operations described below:

$$|\psi_1^i\rangle = \prod_{j=1}^{n} 2\mathrm{XOR}_{p_j^i u_2 m_j} |\psi_0^i\rangle. \tag{5}$$

This simply copies pattern p^i into the memory register of the processing term, identified by $|u_2\rangle = |1\rangle$.

$$|\psi_2^i\rangle = \prod_{j=1}^{n} \mathrm{NOT}_{m_j} \mathrm{XOR}_{p_j^i m_j} |\psi_1^i\rangle,$$
$$|\psi_3^i\rangle = \mathrm{nXOR}_{m_1 \ldots m_n u_1} |\psi_2^i\rangle. \tag{6}$$

The first of these operations makes all qbits of the memory register $|1\rangle$'s when the contents of the pattern and memory registers are identical, which is exactly the case only for the processing term. Together, these two operations change the first utility qbit u_1 of the processing term to a $|1\rangle$, leaving it unchanged for the stored patterns term.

$$|\psi_4^i\rangle = CS_{u_1 u_2}^{p+1-i} |\psi_3^i\rangle. \tag{7}$$

This is the central operation of the storing algorithm. It separates out the new pattern to be stored, already with the correct normalization factor.

$$|\psi_5^i\rangle = \mathrm{nXOR}_{m_1 \ldots m_n u_1} |\psi_4^i\rangle,$$
$$|\psi_6^i\rangle = \prod_{j=n}^{1} \mathrm{XOR}_{p_j^i m_j} \mathrm{NOT}_{m_j} |\psi_5^i\rangle. \tag{8}$$

These two operations are the inverse of Eqs. (6) and restore the utility qbit u_1 and the memory register m to their original values. After these operations one has

$$|\psi_6^i\rangle = \frac{1}{\sqrt{p}} \sum_{k=1}^{i} |p^i; 00; p^k\rangle + \sqrt{\frac{p-i}{p}} |p^i; 01; p^i\rangle. \tag{9}$$

With the last operation,

$$|\psi_7^i\rangle = \prod_{j=n}^{1} 2\mathrm{XOR}_{p_j^i u_2 m_j} |\psi_6^i\rangle, \tag{10}$$

one restores the third register m of the processing term, the second term in Eq. (9) above, to its initial value $|0_1, \ldots, 0_n\rangle$. At this point one can load a new pattern into register p and go through the same routine as just described. At the end of the whole process, the m register is exactly in state $|M\rangle$, Eq. (3).

Assume now one is given a binary input i, which might be, e.g., a corrupted version of one of the patterns stored in the memory. The first step of the information recall process is to make a copy of the memory $|M\rangle$ to be used in the retrieval algorithm described below. Because of the no-cloning theorem [11], this cannot be done deterministically (i.e., using only unitary operations); a faithful copy of $|M\rangle$ can be obtained only with a probabilistic cloning machine [12]. I thus assume the availability of a probabilistic cloning machine for which $|M\rangle$ is one of the set of linearly independent states that can be copied.

The retrieval algorithm requires also three registers. The first register i of n qbits contains the input pattern; the second register m, also of n qbits, contains the memory $|M\rangle$; finally there is a single qbit control register c initialized to the state $(|0\rangle + |1\rangle)/\sqrt{2}$. The full initial quantum state is thus

$$|\psi_0\rangle = \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} |i_1, \ldots, i_n; p_1^k, \ldots, p_n^k; 0\rangle + \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} |i_1, \ldots, i_n; p_1^k, \ldots, p_n^k; 1\rangle. \tag{11}$$

I now apply to it the following combination of quantum gates:

$$|\psi_1\rangle = \prod_{k=1}^{n} \mathrm{NOT}_{m_k} \mathrm{XOR}_{i_k m_k} |\psi_0\rangle, \tag{12}$$

where, as before, the subscripts on the gates refer to the qbits on which they are applied. As a result of this, the memory register qbits are in state $|1\rangle$ if i_j and p_j^k are identical and $|0\rangle$ otherwise:

$$|\psi_1\rangle = \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} |i_1, \ldots, i_n; d_1^k, \ldots, d_n^k; 0\rangle + \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} |i_1, \ldots, i_n; d_1^k, \ldots, d_n^k; 1\rangle, \tag{13}$$

where $d_j^k = 1$ if and only if $i_j = p_j^k$ and $d_j^k = 0$ otherwise.

Consider now the following Hamiltonian:

$$\mathcal{H} = (d_H)_m \otimes (\sigma_3)_c\,, \qquad (d_H)_m = \sum_{k=1}^{n} \left(\frac{\sigma_3 + 1}{2}\right)_{m_k}, \tag{14}$$

where σ_3 is the third Pauli matrix. $\mathcal{H}$ measures the number of 0's in register m, with a plus sign if c is in state $|0\rangle$ and a minus sign if c is in state $|1\rangle$. Given how I have prepared the state $|\psi_1\rangle$, this is nothing else than the number of qbits which are different in the input and memory registers i and m. This quantity is called the Hamming distance and represents the (squared) Euclidean distance between two binary patterns.

Every term in the superposition (13) is an eigenstate of $\mathcal{H}$ with a different eigenvalue. Applying thus the unitary operator $\exp(i\pi\mathcal{H}/2n)$ to $|\psi_1\rangle$ one obtains

$$\begin{aligned} |\psi_2\rangle &= \exp\left(i\,\frac{\pi}{2n}\,\mathcal{H}\right)|\psi_1\rangle\,, \\ |\psi_2\rangle &= \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} \exp\left[i\,\frac{\pi}{2n}\,d_H(i,p^k)\right] |i_1,\ldots,i_n; d_1^k,\ldots,d_n^k; 0\rangle \\ &\quad + \frac{1}{\sqrt{2p}} \sum_{k=1}^{p} \exp\left[-i\,\frac{\pi}{2n}\,d_H(i,p^k)\right] |i_1,\ldots,i_n; d_1^k,\ldots,d_n^k; 1\rangle\,, \end{aligned} \tag{15}$$

where $d_H(i,p^k)$ denotes the Hamming distance between the input i and the stored pattern p^k.

In the final step I restore the memory gate to the state $|M\rangle$ by applying the inverse transformation to Eq. (12) and I apply the Hadamard gate (1) to the control qbit, thereby obtaining

$$\begin{aligned} |\psi_3\rangle &= H_c \prod_{k=n}^{1} \mathrm{XOR}_{i_k m_k} \mathrm{NOT}_{m_k} |\psi_2\rangle\,, \\ |\psi_3\rangle &= \frac{1}{\sqrt{p}} \sum_{k=1}^{p} \cos\frac{\pi}{2n}\,d_H(i,p^k)\,|i_1,\ldots,i_n; p_1^k,\ldots,p_n^k; 0\rangle \\ &\quad + \frac{1}{\sqrt{p}} \sum_{k=1}^{p} \sin\frac{\pi}{2n}\,d_H(i,p^k)\,|i_1,\ldots,i_n; p_1^k,\ldots,p_n^k; 1\rangle\,. \end{aligned} \tag{16}$$

This concludes the deterministic part of the information retrieval process. At this point one needs a measurement of the control qbit c. The probabilities for this to be in states $|0\rangle$ and $|1\rangle$ are given by the expressions

$$P(|c\rangle = |0\rangle) = \sum_{k=1}^{p} \frac{1}{p} \cos^2\left(\frac{\pi}{2n}\,d_H(i,p^k)\right), \tag{17}$$

$$P(|c\rangle = |1\rangle) = \sum_{k=1}^{p} \frac{1}{p} \sin^2\left(\frac{\pi}{2n}\,d_H(i,p^k)\right). \tag{18}$$

If the input pattern is very different from all stored patterns, one has a high probability of measuring $|c\rangle = |1\rangle$. On the contrary, an input pattern close to all stored patterns leads to a high probability of measuring $|c\rangle = |0\rangle$. One can thus set a threshold T: if T repetitions of the retrieval algorithm all lead to a measurement $|c\rangle = |1\rangle$ one classifies the input i as nonrecognized. If one gets a measurement $|c\rangle = |0\rangle$ before the threshold is reached, instead, one classifies the input i as recognized and one can proceed to a measurement of the memory register to identify it. This measurement yields pattern p^k with probability

$$P(p^k) = \frac{1}{pP(|c\rangle = |0\rangle)} \cos^2\left(\frac{\pi}{2n}\,d_H(i,p^k)\right). \tag{19}$$

This probability is peaked around those patterns which have the smallest Hamming distance to the input. The highest probability of retrieval is thus realized for that (those) pattern(s) which is (are) most similar to the input.

What about the efficiency of this information retrieval mechanism? Contrary to any classical counterpart, this efficiency depends here on two features: the threshold T determining recognition and the shape of the probability distribution in Eq. (19), determining the identification. The threshold T should be optimally chosen according to the probabilities in Eqs. (17) and (18) and depends thus on the distribution of the stored patterns. Indeed, the probability of recognition is determined by comparing (squared) cosines and sines of the distances to the stored patterns. It is thus clear that the worst case for recognition is the situation in which there is an isolated pattern, with the remaining patterns forming a tight cluster spanning all the largest distances to the first one. Let me suppose that $p = O(n^x)$, $x \ll n$, and assume for simplicity that $p = 1 + \sum_{k=0}^{x} \binom{n}{k}$ and the distribution is such that exactly all patterns of distances $d_H = n, n-1, \ldots, n-x$ to one isolated pattern are stored. If one presents exactly this isolated pattern as input, one of the (squared) cosines in Eq. (17) is 1, while the rest all take the smallest possible values, giving

$$P(|c\rangle = |0\rangle) > \frac{1}{p} + \frac{\pi^2}{4n^2}\,. \tag{20}$$

In order to have the best recognition efficiency also in this worst case, one should therefore choose the threshold

$T = O(n)$ for $x = 1$ and $T = O(n^2)$ for $n \gg x \geq 2$. While this entails a large number of repetitions, it is still polynomial in the number n of qbits and thus tractable. Note also that the required threshold diminishes when the number of stored patterns becomes very large, since, in this case, the distribution of patterns becomes necessarily more homogeneous. Indeed, for the maximal number of stored patterns $p = 2^n$ one has $P(|c\rangle = |0\rangle) = 1/2$ and the recognition efficiency becomes also maximal, as it should be. In the general case one can initially estimate the p recognition probabilities of the patterns by setting $i = p^k$ for $k = 1, \dots, p$ in Eq. (17). Letting $P_{\min}$ be the smallest of these, one can once and for all choose the threshold T of this memory as the nearest integer to $1/P_{\min}$. I do not discuss here a possible quantum speedup of this calculation since the main point of the present Letter is the exponential storage capacity with retrieval of noisy inputs.

While the recognition efficiency depends on comparing (squared) cosines and sines of the same distances in the distribution, the identification efficiency of Eq. (19) depends on comparing the (squared) cosines of the different distances in the distribution. Specifically, it is best when one of the distances is zero, while all others are as large as possible, such that the probability of retrieval is completely peaked on one pattern. As a consequence, the identification efficiency is best when the recognition efficiency is worst and vice versa.

Having described at length the information retrieval mechanism for complete but possibly corrupted patterns, it is easy to incorporate also incomplete ones. To this end assume that only $q < n$ qbits of the input are known and let me denote these by the indices $\{k_1, \dots, k_q\}$. After assigning the remaining qbits randomly, there are two possibilities. One can treat just the resulting complete input as a noisy one and proceed as above or, better, one can limit the operator $(d_H)_m$ in the Hamiltonian (14) to

$$(d_H)_m = \sum_{i=1}^{q} \left(\frac{\sigma_3 + 1}{2} \right)_{m_{ki}}, \qquad (21)$$

so that the Hamming distances to the stored patterns are computed on the basis of the known qbits only. After this the pattern recall process continues exactly as described above. This second possibility has the advantage that it does not introduce random noise in the similarity measure, but it has the disadvantage that the operations of the memory have to be adjusted to the inputs.

This brings me to the last point, the feasibility of the described algorithms. In this context I point out that, in addition to the standard NOT, H (Hadamard), XOR, 2XOR (Toffoli), and nXOR gates [2] I have introduced only the two-qbit gates CS^i in Eq. (2) and the unitary operator $\exp(i\pi\mathcal{H}/2n)$. It remains thus to show only that this latter can be realized by simple gates involving few qbits. To this end I introduce the single-qbit gate

$$U = \begin{pmatrix} \exp(i\frac{\pi}{2n}) & 0 \\ 0 & 1 \end{pmatrix}, \qquad (22)$$

and the two-qbit controlled [2] gate

$$CU^{-2} = |0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes U^{-2}. \qquad (23)$$

It is then easy to check that $\exp(i\pi\mathcal{H}/2n)$ can be realized as follows:

$$\exp\left(i\,\frac{\pi}{2n}\,\mathcal{H} \right) |\psi_1\rangle = \prod_{i=1}^{n} (CU^{-2})_{cm_i} \prod_{j=1}^{n} U_{m_j} |\psi_1\rangle, \qquad (24)$$

where c is the control qbit in the first series of gates. Essentially, this means that one implements first $\exp(i\pi d_H/2n)$ and then one corrects by implementing $\exp(-i\pi d_H/n)$ on that part of the quantum state for which the control qbit $|c\rangle$ is in state $|1\rangle$. This completes the proof of feasibility.

It remains to point out that the information retrieval algorithm can be, in principle, generalized by substituting the Hamiltonian (14) with

$$\mathcal{H} = [f(d_H)]_m \otimes (\sigma_3)_c, \qquad (25)$$

where f is any function satisfying $f(0) = 0$ and $f(n) = n$. Such a generalization would above all have an influence on the identification efficiency by changing the shape of the probability distribution on the memory, which can be made narrower around the input. One can also give different weights to different qbits by introducing a nontrivial metric. The only restriction on all these generalizations is, as always, the feasibility of the resulting unitary evolution.

*Email address: ca.trugenberger@bluewin.ch

[1] For a review, see A. Steane, Rep. Prog. Phys. **61**, 117 (1998).

[2] For a review, see A. O. Pittenger, *An Introduction to Quantum Computing Algorithms* (Birkhäuser, Boston, 2000).

[3] P. W. Shor, SIAM J. Comput. **26**, 1484 (1997).

[4] L. Grover, Phys. Rev. Lett. **79**, 325 (1997).

[5] For a review, see B. Müller and J. Reinhardt, *Neural Networks* (Springer-Verlag, Berlin, 1990); T. Kohonen, *Self-Organization and Associative Memory* (Springer-Verlag, Berlin, 1984).

[6] J. J. Hopfield, Proc. Natl. Acad. Sci. U.S.A. **79**, 2554 (1982).

[7] B. Kosko, IEEE Trans. Syst. Man Cybern. **18**, 49 (1988).

[8] See, e.g., M. Mezard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond* (World Scientific, Singapore, 1987).

[9] See, e.g., A. Barenco, C. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995).

[10] D. Ventura and T. Martinez, Found. Phys. Lett. **12**, 547 (1999).

[11] W. Wootters and W. Zurek, Nature (London) **299**, 802 (1982).

[12] L.-M. Duan and G.-C. Guo, Phys. Rev. Lett. **80**, 4999 (1998).