LETTER Communicated by Jerome Feldman and Peter Thomas

Memorization and Association on a Realistic Neural Model

Leslie G. Valiant
valiant@deas.harvard.edu
Division of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, U.S.A.

A central open question of computational neuroscience is to identify the data structures and algorithms that are used in mammalian cortex to support successive acts of the basic cognitive tasks of memorization and association. This letter addresses the simultaneous challenges of realizing these two distinct tasks with the same data structure, and doing so while respecting the following four basic quantitative parameters of cortex: the neuron number, the synapse number, the synapse strengths, and the switching times. Previous work has not succeeded in reconciling these opposing constraints, the low values of synapse strengths that are typically observed experimentally having contributed a particular obstacle. In this article, we describe a computational scheme that supports both memory formation and association and is feasible on networks of model neurons that respect the widely observed values of the four quantitative parameters. Our scheme allows for both disjoint and shared representations. The algorithms are simple, and in one version both memorization and association require just one step of vicinal or neighborly influence. The issues of interference among the different circuits that are established, of robustness to noise, and of the stability of the hierarchical memorization process are addressed. A calculus therefore is implied for analyzing the capabilities of particular neural systems and subsystems, in terms of their basic numerical parameters.

1 Introduction

We consider four quantitative parameters of a neural system that together constrain its computational capabilities: the number of neurons, the number of neurons with which each neuron synapses, the strength of synaptic connections, and the speed of response of a neuron. The typical values that these parameters are believed to have in mammalian cortex appear to impose extremely severe constraints. We believe that it is for this reason that computationally explicit mechanisms for realizing multiple cognitive tasks simultaneously on models having these typical cortical parameters have not been previously offered.

Estimates of these four cortical parameters are known for several systems. The number of neurons in mouse cortex has been estimated to be 1.6×10^7, while the corresponding estimate is in the region of 10^{10} for humans (Braitenberg & Schuz, 1998). There also exist estimates of the number of neurons in different parts of cortex and in related structures such as the hippocampus and olfactory bulb.

The number of neurons with which each neuron synapses, which we shall call the degree, is a little harder to measure. However, it is considered that the effect of multiple synapsing between pairs of neurons is small and therefore that this degree is close to the total number of synapses per neuron, which has been estimated to be 7800 in mouse cortex and in the 24,000 to 80,000 range in humans (Abeles, 1991).

The third parameter, the synapse strength, presents a still more complex set of issues. The most basic question here is how many of a neuron's neighbors need to be sending an action potential in order to create an action potential in the neuron. Equivalently, the contribution of each synapse, the excitatory presynaptic potential, can be measured in millivolts and the fraction that this constitutes of the threshold voltage that needs to be overcome evaluated. While some moderately strong synapses have been recorded (Thomson, Deuchars, & West, 1993; Markram & Tdodyks, 1996a; Ali, Deuchars, Pawelzik, & Thomson, 1998), the average value is believed to be weak. The effective fraction of the threshold that each neighbor contributes, has been estimated (Abeles, 1991) to be in the range 0.003 to 0.2. In other words, it is physiologically quite possible that cognitive computations are characterized by a very small and hard-to-observe fraction of synapses that contribute above this range, perhaps even up to the threshold fraction of 1.0. However, there is no experimental confirmation of this to date, and for that reason, this article addresses the possibility that at least some neural systems work entirely with synapses whose strengths are some orders of magnitude smaller.

The time it takes for a neuron to complete a cycle of causing an action potential in response to action potentials in its presynaptic neurons has been estimated as being in the 1 to 10 millisecond range. Since mammals can perform significant cognitive tasks in 100 to 200 milliseconds, algorithms that take a few dozen parallel steps must suffice.

The central technical contribution of this article is to show that two basic computational problems, memory formation and association, can be implemented consistently in model neural systems that respect all of the above-mentioned numerical parameters. The first basic function, JOIN, implements memory formation of a new item in terms of two established items: if two items A and B are already represented in the neural system, the task of JOIN is to modify the circuit so that at subsequent times, there is the representation of a new item C that will fire if and only if the representations of both A and B are firing. Further, the representation of C is an "equal citizen" with those of A and B for the purposes of subsequent calls of JOIN. JOIN is intended as the basic form of memorization of an item and incorporates the idea that such memorization has to be indexed in terms of the internal representations of items already represented. The second function, LINK, implements association. If two "items" A and B are already represented in the neural system in the sense that certain inputs can cause either of these to fire, the task of LINK is to modify the circuit so that at subsequent times, whenever the representation of item A fires, the modified circuit will cause the representation of item B to fire also.

Implicit in the definitions of both JOIN and LINK is the additional requirement that there be no deleterious interference or side effects. This means that the circuit modifications do not impair the functioning of previously established circuits; that when the newly created circuit executes, no unintended other items fire; and that the intended action of the new circuit cannot be realized in consequence of some unintended condition.

We note that for some neural systems, such as the hippocampus and the olfactory bulb, the question of what items, whether representing location or odors, for example, are being represented has been the subject of some experimental study already. Also for such systems, our four cortical parameters can be measured. We therefore expect that our analysis offers both explanatory and predictive value for understanding such systems. For the parts of cortex that process higher-level functions, the corresponding experimental evidence is more elusive.

In order that our results apply to a wide range of neural systems, we describe computational results for systems within a broad range of realistic parameters. We show that for wide ranges of values of the neuron count between 10^5 and 10^9, and of values of the synapse count or degree between 16 and 10^6, there is a range of values of the synapse strength between .001 and .125 for which both JOIN and LINK can be implemented. Furthermore, this latter range usually includes synaptic strengths that are at the small end of the range. Tables 1 through 4 summarize these data and show, given the values of the neuron count and degree of a neural system, the maximum synapse strength that is sufficient for JOIN and LINK in both disjoint and shared representations. The implied algorithms for LINK take just one step and for JOIN either two steps (see Tables 1 and 2) or also just one step (see Tables 3 and 4). The simplicity of these basic algorithms leaves room for more complex functions to be built on top of them.

We also describe a general relationship among the parameters that holds under some stated assumptions for systems that use the mechanisms described. This relationship $(*)$ states that kn exceeds rd, but only by at most a fixed small constant factor, where $1/k$ is the maximum collective strength of the synapses to any one neuron from any one of its presynaptic neurons, n is the number of neurons, r is the number of neurons that represent a single item, and d is the number of neurons from which each neuron receives synapses.

The essential novel contribution of this article is to show that random graphs have some unexpected powers. In particular, for parameters that have been observed in biology, they allow a method of assigning memory to a new item and also allow for paths, and algorithms for establishing the paths, for realizing associations between items.

There is a long history of studies of random connectivity for neural network models, notably Beurle (1955), Griffith (1963, 1971), Braitenberg (1978), Feldman (1982), and Abeles (1991). In common with such previous studies, ours assumes random interconnections and does not apply to systems where, for example, the connections are strictly topographic. The other component of our approach that also has some history is the study of local representations in neural networks, including Barlow (1972), Feldman (1982), Feldman and Ballard (1982), Shastri and Ajjanagadde (1993), and Shastri (2001). The question of how multiple cognitive functions can be realized simultaneously using local representations and random connections has been pursued by Valiant (1988, 1994).

Our central subject matter is the difficulty of computing flexibly on sparse networks where nodes are further frustrated in having influence on others by the weakness of the synapses. This difficulty has been recognized most explicitly in the work of Griffith (1963) and Abeles (1991). Griffith suggests communication via chains that consist of sets of k nodes chained together so that each member of each set of k nodes is connected to each member of the next set in the chain. If the synaptic strength of each synapse is $1/k$, then a signal can be maintained along the chain. Abeles suggests a more general structure, which he calls a synfire chain, in which each set has $h \geq k$ nodes and each node is connected to k of the h nodes in the next set. He shows that for some small values of k, such chains can be found somewhere in suitably dense such networks.

The goals of this article impose multiple constraints for which these previous proposals are not sufficient. For example, for realizing associations, we want that between any pair of items, there is a potential chain of communication. In other words, these chains have to exist from anywhere to anywhere in the network rather than just somewhere in the network. A second constraint is that we want explicit computational mechanisms for enabling the chain to be invoked to perform the association, and the passive existence of the chain in the network is not enough. A third requirement is that for memory formation, we need connectivity of an item to two others.

Some readers may choose to view this article as one that solves a communication or wiring problem, and not a computational one. This view is partly justified since once it is established that the networks have sufficiently flexible interaction capabilities, the mechanisms required at the neurons are computationally very simple. For readers who wish to investigate more rigorously what *simple* means here, we have supplied a section that goes into more details. The model of computation used there is the neuroidal model (Valiant, 1994), which was designed to capture the communication capabilities and limitations of cortex as simply as possible. It assumes only the simplest timing and state change mechanisms for neurons so that there can be no doubt that neurons are capable of doing at least that much. Demonstrating that some previously mysterious task can be implemented even on this simple model therefore has explanatory power for actual neural systems.

The neuroidal model was designed to be more generally programmable than its predecessors and hence to offer the challenge of designing explicit computational mechanisms for explicitly defined and possibly multiple cognitive tasks. The contribution of this article may be viewed as that of exhibiting a wide range of new solutions to that model. The previous solutions given for the current tasks were under the direct action hypothesis—the hypothesis that synapses could become so strong that a single presynaptic neuron was enough to cause an action potential in the postsynaptic neuron. Whether this hypothesis holds for neural systems that perform the relevant cognitive tasks is currently unresolved. In contrast, the mechanisms described here are in line with synaptic strength values that have been widely observed and generally accepted.

This letter pursues a computer science perspective. In that field, it is generally found, on the positive side, that once one algorithm has been discovered for solving a computational problem within specified resource bounds, many others often follow. On the other hand, on the negative side, it is found that the resource bounds on computation can be very severe. For example, for the NP-complete problem of satisfiability (Cook, 1971; Papadimitriou 1994) of boolean formulas with n occurrences of literals, no algorithm for solving all instances of it in $2^{f(n)}$ steps is known for any function $f(n)$ growing more slowly than linear in n. If a device were found that could solve this problem faster, then a considerable mystery would be created: the device would be using some mechanism that is not understood. Neuroscience has mysteries of the same computational nature and needs to resolve them. This letter aims at making one of these mysteries concrete and to resolve it.

2 Graph Theory

We consider a random graph G with n vertices (Bollobas, 2001). From each vertex, there is a directed edge to each other vertex with probability p, so that the expected number of nodes to which a node is connected is $d = p(n-1)$. In this model, a vertex corresponds to a neuron, and a directed edge from one vertex to another models the synapse between the presynaptic neuron and the postsynaptic neuron. Such a model makes sense for neural circuits that are richly interconnected. A variant of the model is that of a bipartite graph where the vertex set can be partitioned into two subsets V_1 and V_2, such that every edge is directed from a V_1 vertex to a V_2 vertex. This would be appropriate for modeling the connections from one area of cortex to a second, possibly distant, area. The analyses we give for JOIN and LINK apply equally to both variants. We shall use $d = pn$ in the analysis, which is exact for the bipartite case, and a good approximation for the other case for large n.

In general, to obtain rigorous results about random graphs, we take the view that for the fixed nodes under consideration, the edges are present or not, each with probability p independent of each other. It is convenient for analysis to view the edges as being generated in that manner afresh rather than as fixed at some previous time.

We assume that the maximum synaptic strength is $1/k$ of the threshold, for some integer k. In the graph theoretic properties, we shall therefore always need to find at least k presynaptic neighbors to model the k presynaptic neurons that need to be active to make the neuron in question active.

Finally, we shall model the representation of an item in cortex by a set of about r neurons, where r is the *replication factor*. In general, such an item will be considered to be recognized if essentially all the constituent neurons are active. In general, different items will be represented by different numbers of neurons, though of the same order of magnitude. We do not try to ensure that they are all represented by exactly r. However, once an item is represented by some r' neurons, then it makes sense to assert that if no more than $r'/2$ of its members are firing, then the item has not been recognized.

We call a representation *disjoint* or *shared*, respectively, depending on whether the sets that represent two distinct items need, or need not, be disjoint. In disjoint representations, clearly, no more than n/r items can be represented, while shared representations allow for many more, in principle.

2.1 Memory Formation for Disjoint Representations. The JOIN property is the following. Given values of n, d, and k, which are the empirical parameters of the neural system, we need to show that the following holds. *Given two subsets of nodes A and B of size r, the number of nodes to which there are at least k edges directed from A nodes and also at least k edges directed from B nodes has expected value of r.* The vertices that are so connected will represent the new item C. The above-mentioned property ensures that the representation of C will be made to fire by causing the representation of either one of A or B to fire. The required network is illustrated in Figure 1. We want C to be an equal citizen as much as possible with A and B for the purposes of further calls of JOIN. We ensure this by requiring that the expected number of nodes that represent C is the same as the number of those that represent A and B.

In general, we denote by $\mathcal{B}(r, p, k)$ the probability that in r tosses of a coin that comes up heads with probability p and tails with probability $1-p$, there will be k or more heads. This quantity is equal to the sum for j ranging from k to r of the value of $\mathcal{T}(r, p, j) = (r!/(j!(r-j)!)p^j(1-p)^{r-j}$. For constructing our tables, we compute such terms to double precision using an expansion

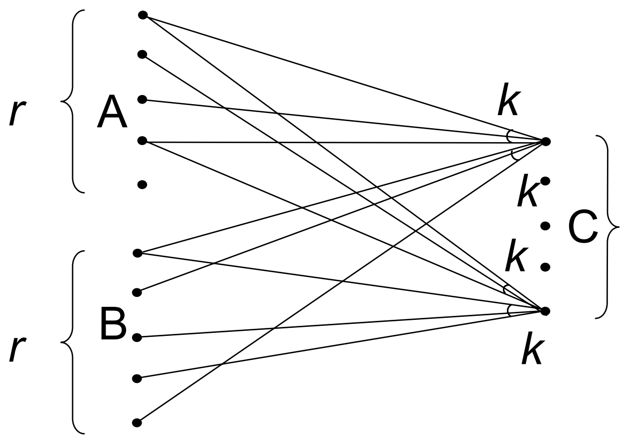

Figure 1: Graph-theoretic structure needed for the two-step algorithm for disjoint representations for the memorization of the conjunction of items at A and B. For shared representations, the sets A and B may intersect. For the one-step algorithm, there is a bound k_m on the total number of edges coming from A and B rather than bounds on A and B separately.

for the logarithm of the factorial or gamma function (Abramowitz & Stegun, 1964).

We now consider the JOIN property italicized above. For each vertex u in the network, the probability that it has at least k edges directed toward it from the r nodes of A is $\mathcal{B}(r, p, k)$, since each vertex of A can be regarded as a coin toss with probability p of heads (i.e., of being connected to u) and we want at least k successes. The same holds for the nodes of B. Hence the probability of a node being connected in this way to both A and B is $p' = (\mathcal{B}(r, p, k))^2$, and hence the expected number of vertices so connected is n times this quantity. The stated requirement on the JOIN property therefore is that the following be satisfied:

$$n(\mathcal{B}(r, p, k))^2 = r. \tag{2.1}$$

This raises the important issue of stability. Even if the numbers of nodes assigned to A and B are both exactly r, this process will assign r nodes to C only in expectation. How stable is this process if such memorization operations are performed in sequence, with previously memorized items forming the A and B items of the next memorization operation? Fortunately, it is easy to see that this process gets more and more stable as r increases. The argument for relationship 2.1 showed that the number of nodes assigned to C is a random variable with a binomial distribution defined as the number of successes in n trials where the probability of success in each trial is p'.

This distribution therefore has mean np', as already observed, and variance $np'(1-p')$. The point is that this variance is close to the mean $r = np'$ if p' is small relative to 1, and then the standard deviation, which is the square root of the variance, is approximately $\sqrt{r}$. Hence, the standard deviation as a fraction of the mean decreases as $1/\sqrt{r}$ as the mean $np' = r$ increases. For the ranges of values that occur typically in this article, such as r equal to $10^3, 10^4$, or even much larger, this $\sqrt{r}$ standard deviation will be a small fraction of r, and hence one can expect the memorization process to be stable for many stages. Thus, for stability, the large k large r cases considered in this article are much more favorable than the $k = 1$ case considered in Valiant (1994) with $r = 50$. For the latter situation, some analysis and suggested ways of coping with the more limited stability were offered in Valiant (1994) and Gerbessiotis (2003).

If fewer than a half of the representatives of an item are firing, we regard that item as not being recognized. As a side-effect condition, we therefore want that if no more than a half of one of A or B is active, then the probability that more than a half of C is active is negligible. Since we cannot control the size of C exactly, we ensure the condition that at most half of C be active by insisting that at most a much smaller fraction of r, such as $r/10$, be active. The intention is that $r/10$ will be smaller than a half of C even allowing for the variations in the size of C after several stages of memory allocation. This gives

$$\mathcal{B}(n, \mathcal{B}(r/2, p, k)\mathcal{B}(r, p, k), r/10) \sim 0. \tag{2.2}$$

The second side-effect condition we impose is related to the notion of capacity, or the number of items that can be stored. To guarantee large capacity, we need an assurance that the A $\wedge$ B nodes allocated will not be caused to fire if a different conjunction is activated. The bad case is if the second conjunction involves one of A or B, say A, and another item D different from B. If the node sets allocated to A $\wedge$ B and not to A $\wedge$ D is of size at least $2r/3$, then we will consider there to be no interference since if A $\wedge$ B is of size at most $4r/3$, then the firing of A $\wedge$ D will cause fewer than half of the nodes of A $\wedge$ B to fire.

The probability that a node receives k inputs from B and k from A, but fewer than k from D, is $p' = (1 - \mathcal{B}(r, p, k))(\mathcal{B}(r, p, k))^2$, and we want that the number of nodes that are so allocated to A $\wedge$ B but not to A $\wedge$ D to be at least $2r/3$. Hence, we want

$$\mathcal{B}(n, p', 2r/3) \sim 1. \tag{2.3}$$

2.2 Association for Disjoint Representations. We now turn to the LINK property, which ensures that B can be caused to fire by A via an intermediate set of "relay" neurons: *Given two sets A and B of r nodes, for each B vertex u with high probability, the following occurs: the number of (relay) vertices from which*

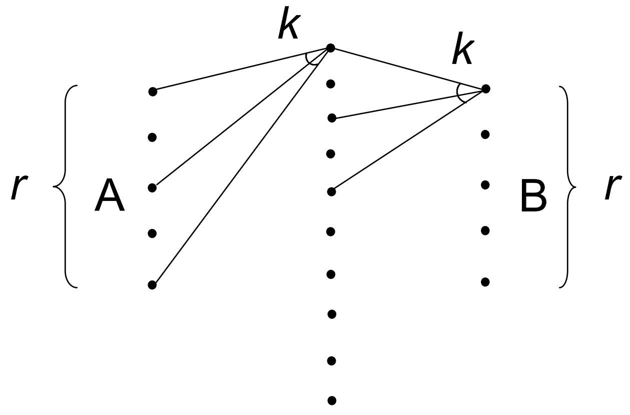

Figure 2: Graph-theoretic structure needed for the algorithm for establishing an association of the item at A to the item at B.

there is a directed edge to u and to which there are at least k edges directed from A nodes is at least k. We shall call this probability Y. This property ensures that each neuron u that represents B will be caused to fire with high probability if the A representation fires. This property is illustrated in Figure 2.

For the LINK property, we note that the probability of a vertex having at least k connections from A and also having a connection to B vertex u is $p\mathcal{B}(r, p, k)$. We need the number of such nodes to be at least k with high probability, or in other words that

$$Y = \mathcal{B}(n, p\mathcal{B}(r, p, k), k) \sim 1. \tag{2.4}$$

As a side-effect condition, we need that if at most half of A fire, then with high probability, fewer than half of B should fire. We approximate this quantity by assuming independence for the various u:

$$\mathcal{B}(r, \mathcal{B}(n, p\mathcal{B}(r/2, p, k), k), r/2) \sim 0. \tag{2.5}$$

As a second side-effect condition, we consider the probability that a third item C for which no association with B has been set up by a LINK operation will cause B to fire because some relay nodes are shared with A. Further, we make this more challenging by allowing there to have been t, rather than just one, association with B previously set up, say from $\mathrm{A}_1, \ldots, \mathrm{A}_t$. Now, the probability that a node will act as a relay node from A_1 to a fixed node u in B is $p\mathcal{B}(r, p, k)$. For t such previous associations, the probability that a node acts as a relay for at least one of the A_i is $p' = p(1-(1-\mathcal{B}(r, p, k))^t)$. If we require that these nodes be valid relay nodes from item C also, then this probability gets multiplied by another factor of $\mathcal{B}(r, p, k)$ since C is disjoint from $\mathrm{A}_1, \ldots, A_t$.

Then the side-effect requirement becomes that $p'' = \mathcal{B}(n, p'\mathcal{B}(r, p, k), k)$, the probability of there being at least k relay nodes for u is so small that it is unlikely that a large fraction, say at least $r/2$, of the B nodes are caused to fire by C. We approximate this quantity also by making the assumption of independence for the various u:

$$\mathcal{B}(r, p'', r/2) \sim 0. \tag{2.6}$$

2.3 Memory Formation for Shared Representations. By *shared representation*, we mean a representation where each neuron can represent more than one item. There is no longer a distinction between nodes that have and those that have not been allocated. The items each node will be assigned by JOIN are already determined by the network connections without any training process being necessary. The actual meaning of the items that will be memorized at a node will, of course, depend on the meanings assigned by a different process, such as the hard wiring of certain sensory functions to some input nodes.

The model here is that an item is represented by r neurons, randomly chosen. The expected intersection of two such sets then is of size r^2/n. We can recompute the relations corresponding to equations 2.1 to 2.6 under this assumption. For simplicity, we shall consider only the case in which for JOIN, the neurons of A and B are in one area, and those of C in another, and for LINK the neurons of A, B and the relay nodes are in three different areas, respectively. Then, if for simplicity we make the assumption that the intersection is of size exactly r', the closest integer to r^2/n, then relation 2.1 for JOIN becomes:

$$n\{\mathcal{B}(r', p, k) + \Sigma_{(i=0:k-1)}[\mathcal{T}(r', p, i)(\mathcal{B}(r - r', p, k - i))^2]\} = c_1 r, \tag{2.1$'$}$$

where i in the summation indexes the number of connections from the intersection of A and B. We need to show that c_1 is close to one.

Equation 2.1$'$ we adapt from equation 2.2 to be the following:

$$\mathcal{B}(n, p', r/10) \sim 0, \tag{2.2$'$}$$

where $p' = (\mathcal{B}(r', p, k) + \Sigma_{(i=0,\ldots,k-1)}[\mathcal{T}(r', p, i)\mathcal{B}(r - r', p, k - i)\mathcal{B}(r/2 - r', p, k - i)])$, where i in the summation indexes the number of connections from a neuron that is in both A and B, and $r' = r^2/n$.

We adapt equation 2.3$'$ from equation 2.3 by considering the case that the intersections $A \cap B$, $A \cap D$, and $B \cap D$ all have their expected sizes $r' = r^2/n$, and that $A \cap B \cap D$ has its expected size $r'' = r^3/n^2$. For a fixed node in C, we shall denote the numbers of connections to these four intersections by $i + m$, $j + m$, $l + m$, and m, respectively, in the summation below. Then the probability of a node being allocated to $A \wedge B$ and not to $A \wedge D$ is lower bounded by the following quantity p', where $r^\wedge = r' - r''$:

$$\begin{aligned}&\Sigma_{(i=0:k)}\mathcal{T}(r^\wedge, p, i)\Sigma_{(j=0:k-i)}\mathcal{T}(r^\wedge, p, j)\Sigma_{(l=0:k-i-j)}\mathcal{T}(r^\wedge, p, l)\\ &\qquad\Sigma_{(m=0:k-i-j-l)}\mathcal{T}(r'', p, m)\\ &\qquad[\mathcal{B}(r-2r'+r'', p, k-i-j-m)\mathcal{B}(r-2r'+r'', p, k-j-l-m)\\ &\qquad(1-\mathcal{B}(r-2r'+r'', p, k-i-l-m))].\end{aligned}$$

(Note that to allow for terms in which the intersection of A and B have more than k connections to the C node, we need to interpret the first term $\mathcal{T}(r^\wedge, p, i)$ in the above expression for the particular value $i = k$ to mean $\mathcal{B}(r^\wedge, p, k)$.) Then the relationship we need is

$$\mathcal{B}(n, p', 2r/3) \sim 1. \tag{2.3$'$}$$

2.4 Association for Shared Representations. Equations 2.4′ and 2.5′ in the shared coding are identical to equations 2.4 and 2.5 since we are assuming here, for simplicity, that in an implementation of LINK between A and B, the neurons representing A, B, and the relay nodes are from three disjoint sets.

Equation 2.6′ will correspond to equation 2.6 in the special case of $t = 1$. For a fixed node in B, the probability of a node u being a relay node to it and having k edges coming from both A and C is

$$p' = p(\mathcal{B}(r', p, k) + \Sigma_{(i=0,\ldots,k-1)}[\mathcal{T}(r', p, i)(\mathcal{B}(r-r', p, k-i))^2]).$$

The probability that there are at least k of these is $p'' = \mathcal{B}(n, p', k)$. We want the probability that at least half of the members of B have such sets of at least k relay nodes to be small. We approximate this quantity by assuming independence for the various u:

$$\mathcal{B}(r, p'', r/2) \sim 0. \tag{2.6$'$}$$

2.5 One-step Memory Formation for Shared Representations. The algorithm for memorization implied above requires A and B to be active at different time instants and therefore requires the memorization process to take two steps. Here we shall discuss a process by which memorization can be achieved in one step. For brevity, we shall consider one-step only for the shared representation case. The results are presented in Tables 3 and 4.

The main advantage of these one-step algorithms is that the algorithms become even simpler and assume even less about the timing mechanism of the model of computation. But one small complication arises: the number of inputs needed to fire a node in the memorization algorithm is now different from that needed in the association algorithm. Instead of having a single parameter k, we shall have two parameters, k_m and k_a, respectively. It turns out that $k_m = 2k_a$ works. This means that we can fix the parameter k of the neurons to be k_m, and then use a weight in the memorization algorithm that is only half as strong as the maximum value allowed.

We now consider the JOIN property. For each vertex u that is a potential representative of C, the probability that it has at least k_m edges directed toward it from the nodes of A ∪ B is now $p' = \mathcal{B}(2r - r', p, k_m)$, provided the intersection of A and B is of size exactly r', the integer closest to the expectation r^2/n. This is because each vertex of A ∪ B may be regarded as a coin toss with probability p of coming up heads (i.e., of being connected to u), and we want at least k_m successes. Hence, the expected number of vertices so connected is n times this quantity. The stated requirement on the JOIN property therefore is that the following be satisfied:

$$n\mathcal{B}(2r - r', p, k_m) - r. \tag{2.1''}$$

Alternatively one could impose some specific probability distribution on the choice of A and B and compute an analog of equation 2.1″ that is precise for that distribution and does not need the assumption that the intersections are of size exactly r'.

If fewer than half of the representatives of an item are firing, we regard that item as not being recognized. As a side-effect condition, we therefore want that if no more than half of one of A or B is active, then the probability that more than half of C is active is negligible. In exact analogy with equations 2.2 and 2.2′, we have that

$$\mathcal{B}(n, \mathcal{B}(3r/2, p, k_m), r/10) \sim 0. \tag{2.2''}$$

Here, $3r/2$ upper bounds the number of firing nodes in A ∪ B if at most $r/2$ are firing in A and all are firing in B, say.

As a second side-effect condition, we again need an assurance that the A ∧ B nodes allocated will not be caused to fire if a different conjunction is activated. Again, a bad case is if the second conjunction is A ∧ D where D is different from B. If the node set allocated to A ∧ B and not to A ∧ D is at least of size $2r/3$, we will consider there to be no interference since if A ∧ B is of size less than $4r/3$ then the firing of A ∧ D will cause fewer than half of the nodes of A ∧ B to fire.

If A, B, and D were disjoint sets of r nodes, then the probability that a node receives k_m inputs from A ∪ B but fewer than $k = k_m$ from A ∪ D would be

$$p' = \Sigma_{(s=0:k-1)}\mathcal{T}(r, p, s)(\mathcal{B}(r, p, k_m - s))(1 - \mathcal{B}(r, p, k_m - s)),$$

where s denotes the number of nodes in A that are connected to that node. We want the number of nodes that are so allocated to A ∧ B but not to A ∧ D to be at least $2r/3$. Hence we would want that

$$\mathcal{B}(n, p', 2r/3) \sim 1. \tag{2.3''}$$

In the event that A, B, and D are not disjoint but randomly chosen sets of r elements, we need equation 2.3$''$ but with a value of p' computed as follows. We assume that the intersections A∩B, A∩D, and B∩D all have their expected sizes $r' = r^2/n$, and that $A \cap B \cap D$ has its expected size $r'' = r^3/n^2$. For a fixed node in C, we shall denote the number of connections to these four intersections by $i+m$, $j+m$, $l+m$, and m, respectively, in the summation below. Then the probability of a node being allocated to $A \wedge B$ and not to $A \wedge D$ is lower-bounded by the following quantity p', where $r^{\wedge} = r' - r''$ and $r^{\#} = r - -2r' + r''$:

$$
\begin{aligned}
p' = {} & \Sigma_{(s=0:k-1)}\mathcal{T}(r^{\#}, p, s)\Sigma_{(i=0:k-s-1)}\mathcal{T}(r^{\wedge}, p, i)\Sigma_{(j=0:k-i-s-1)}\mathcal{T}(r^{\wedge}, p, j) \\
& \Sigma_{(m=0:k-i-j-s-1)}\mathcal{T}(r'', p, m)\Sigma_{(l=0:k-i-j-m-s-1)}\mathcal{T}(r^{\wedge}, p, l) \\
& [(\mathcal{B}(r^{\#}, p, k_m - s - i - j - l - m)) \\
& \ (1 - \mathcal{B}(r^{\#}, p, k_m - s - i - j - l - m))].
\end{aligned}
$$

Here s indexes the number of connections from nodes that are in A but not in B or D.

2.6 One-Step Association with Shared Representations. For the LINK property we simply have relations 2.4, 2.5, and 2.6$'$ with k replaced by k_a:

$$Y = \mathcal{B}(n, p\mathcal{B}(r, p, k_a), k_a) \sim 1, \tag{2.4$''$}$$

$$\mathcal{B}(r, \mathcal{B}(n, p\mathcal{B}(r/2, p, k_a), k_a), r/2) \sim 0, \tag{2.5$''$}$$

and

$$\mathcal{B}(r, p'', r/2) \sim 0, \tag{2.6$''$}$$

where

$$
\begin{aligned}
& p'' = \mathcal{B}(n, p', k), \text{ and} \\
& p' = p(\mathcal{B}(r', p, k_a) + \Sigma_{(i=0,\ldots,k-1)}[\mathcal{T}(r', p, i)(\mathcal{B}(r - r', p, k_a - i))^2]).
\end{aligned}
$$

Since equation 2.6$''$ guarantees only that one previous association to item C will not lead to false associations with C, we also use equation 2.6 to compute an estimate of the maximum number of previous associations that still allow resistance to such false associations.

3 Graph-Theoretic Results

In Tables 1 and 2 we summarize the solutions we have found. For each combination of n, d, and k, the table entry gives a value of r that satisfies all six conditions 2.1 to 2.6 to high precision, as well as their equivalents, conditions 2.1′, 2.2′, 2.3′, and 2.6′, for shared representations. (There are essentially only eight equations since equations 2.2′ and 2.6′, subsume equations 2.2 and 2.6, respectively.) For example consider a neural system with $n = 1{,}000{,}000$ neurons where each one is connected to 8192 others on the average and the maximum synaptic strengths are 1/64 of the threshold amount. The entry 8491 found in Table 1 gives the value of r that solves the 10 constraints. It means that if each item is represented by about 8491 neurons, then the graph has the capability of realizing JOIN and LINK using the algorithms to be outlined in later sections. *The central positive result of this article is the existence of entries in the tables for combinations of neuron numbers, synapse strengths, and synapse numbers that are widely observed in neural systems.* We expect that the analysis that underlies the tables offers a basis for a calculus for understanding the algorithms and data structures used in specific systems, such as the hippocampus or the olfactory bulb.

In interpreting the tables the following comments are in order. The solutions were found by solving equation 2.1 using binary search, and discarding any solutions that failed to solve the remaining relations. As a further comment, we note that equation 2.1 and some of the others are defined only for integer values of r, and in our search we therefore imposed the constraint of allowing only such integer values. The values of r are therefore the integer values for which the value of the left-hand side of equation 2.1 is as close as possible to r. We detail the exact integer values as a reminder of this. For all the entries shown, the difference between the two sides of equation 2.1 is at most 1%, except for those labeled $*$, where a difference of 10% is allowed.

The following are some further observations: The case of $k = 1$ in equation 2.4 is known (Valiant, 1994) to give an asymptotic value of $Y \sim 1-1/e = .63\ldots$. In general, values $k = 1, 2$, and 4 violate at least one of either equation 2.2 or 2.5. Most of the entries support equation 2.6 only up to $t = 1$, except for some entries with $k = 8$ or 16 and with some of the higher values of d. Finally, corresponding entries for different values of the neuron count n are in the ratio of the values of n.

We note that for the $k = 1$ case, the analysis in Valiant (1994) relates to the analysis here in the following way. It is observed there that in general for any r and n, the graph density that supports JOIN is too sparse to support LINK with a Y value close to 1. The suggested solution there is to use a graph that is dense enough to support LINK and to have it ignore a random fraction of the connections when implementing JOIN so as to effectively use a sparser graph regime for that purpose. On a separate issue, the earlier analysis did not have any equivalents of the relations 2.2 and 2.5 above. For disjoint representations, where the intention is that in each situation, either all or none of the representatives of an item fires, these conditions might be argued to be too onerous.

Tables 3 and 4 summarize the solutions we have found that support the one-step algorithm. We note that in general, corresponding values of r are a

Table 1: Value of the Replication Factor r That Is the Closest Integer Solution to Equation 2.1 for the Given Values of the Neuron Count n, the Degree d, and the Inverse Synaptic Strength k.

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 100{,}000$ neurons								
$d = 128$								
$d = 256$	1981	5025						
$d = 512$	899	2338	5420					
$d = 1024$	412	1098	2582					
$d = 2048$		519	1238	2749				
$d = 4096$		*247	597	1337	2865			
$d = 8192$		*119	*290	654	1407			
$d = 16{,}384$			*143	*322	*695	1458		
$d = 32{,}768$				^*162	^*347	^*727	*1496	
$d = 65{,}536$							^*753	
$n = 1{,}000{,}000$ neurons								
$d = 128$								
$d = 256$	19,803	50,235						
$d = 512$	8979	23,365	54,181					
$d = 1024$	4105	10,957	25,796					
$d = 2048$	1888	5168	12,353	27,460				
$d = 4096$	^873	2449	5940	13,330	28,607			
$d = 8192$	^406	1165	2866	6491	14,015			
$d = 16{,}384$	^190	^557	1388	3169	6883	14,497		
$d = 32{,}768$		^268	^675	1552	^3388	7161	14,836	
$d = 65{,}536$		^*130	^330	^*763	^*1672	^3545	^7360	
$d = 131{,}072$			^*163	^*378	^*830	^1761	^3660	
$d = 262{,}144$				^*190	^*415		^*1827	
$d = 524{,}288$								
$n = 10{,}000{,}000$ neurons								
$d = 128$								
$d = 256$	198,025	502,339						
$d = 512$	89,777	233,636	541,791					
$d = 1024$	41,033	109,547	257,940					
$d = 2048$	18,868	51,660	123,498	274,571				
$d = 4096$	^8717	24,467	59,368	133,265	286,021			
$d = 8192$	^4043	11,628	28,628	64,866	140,098			
$d = 16{,}384$	^1882	^5542	13,839	31,643	68,761	144,886		
$d = 32{,}768$	^879	^2649	6704	15,464	33,802	71,516	148,248	
$d = 65{,}536$		^1269	^3255	^7570	^16,640	^35,342	^73,463	
$d = 131{,}072$		^610	^1584	^3712	^8202	^17,484	^36,437	^74,842
$d = 262{,}144$		^295	^773	^1824	^4049	^8660	^18,089	
$d = 524{,}288$								
$d = 1{,}048{,}576$								

Notes: Equation 2.1 is accurate to ratio 10^{-2} (but only 10^{-1} if marked by $*$) and Equation 2.4 to 10^{-6}. Equations 2.2, 2.3, 2.5, 2.6, 2.2′, 2.3′, and 2.6′ are accurate to 10^{-6}. For unmarked entries these accuracies are achieved even if the noise rates are 10^{-4} for equations 2.2, 2.3, 2.5, and 2.2′; 10^{-5} for 2.6 and 2.3′; and 10^{-6} for 2.6′. For entries marked with a ^, this accuracy is achieved with the lower noise rate 10^{-6} for all seven equations. Equation 2.1′ is satisfied with constant $1 < c_1 < 1.1$.

Table 2: The Value of the Replication Factor r That Is the Closest Integer Solution to Equation 2.1 for the Given Values of the Neuron Count n, the Degree d, and the Inverse Synaptic Strength k.

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 100{,}000{,}000$ neurons								
$d = 128$								
$d = 256$	1,980,239	5,023,377						
$d = 512$	897,763	2,336,342	5,417,894					
$d = 1024$	410,318	1,095,455	2,579,374					
$d = 2048$	188,664	516,578	1,234,952	2,745,675				
$d = 4096$	^87,153	244,648	593,653	1,332,609	2,860,166			
$d = 8192$	^40,412	116,254	286,244	648,612	1,400,921			
$d = 16{,}384$	^18,797	^55,394	138,351	316,375	687,547	1,448,778		
$d = 32{,}768$	^8766	^26,455	67,001	154,582	337,949	715,063	1,482,369	
$d = 65{,}536$	^4098	^12,660	^32,501	^75,636	^166,314	^353,310	^734,499	
$d = 131{,}072$		^6069	^15,789	^37,053	^81,930	^174,721	^364,217	^748,226
$d = 262{,}144$		^2915	^7681	^18,172	^40,397	^86,468	^180,718	^371,950
$d = 524{,}288$								
$d = 1{,}048{,}576$								

Table 2: *Continued.*

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 1{,}000{,}000{,}000$ neurons								
$d = 128$								
$d = 256$	19,802,377	50,233,759						
$d = 512$	8,977,613	23,363,401	54,178,923					
$d = 1024$	4,103,166	10,954,534	25,793,723					
$d = 2048$	1,886,626	5,165,758	12,349,496	27,456,714				
$d = 4096$	^871,517	2,446,454	5,936,498	13,326,048	28,601,613			
$d = 8192$	^404,099	1,162,518	2,862,401	6,486,077	14,009,157			
$d = 16{,}384$	^187,946	553,914	1,383,469	3,163,694	6,875,400	14,487,695		
$d = 32{,}768$	^87,638	^264,523	669,965	1,545,764	3,379,417	7,150,540	14,823,572	
$d = 65{,}536$	^40,955	^126,564	^324,966	^756,296	^1,633,055	^3,532,995	^7,344,851	
$d = 131{,}072$		^60,656	^154,842	^370,461	^819,214	^1,747,094	^3,642,013	^7,482,072
$d = 262{,}144$		^29,112	^76,758	^181,644	^403,874	^864,553	^1,807,014	^3,719,279
$d = 524{,}288$								
$d = 1{,}048{,}576$								

Notes: Equation 2.1 is accurate to ratio 10^{-2} and equation 2.4 to 10^{-6}. Equations 2.2, 2.3, 2.5, 2.6, 2.2′, 2.3′, and 2.6′ are accurate to 10^{-6}. For unmarked entries, these accuracies are achieved even if the noise rates are 10^{-4} for 2.2, 2.3, 2.5, and 2.2′; 10^{-5} for 2.6 and 2.3′; and 10^{-6} for 2.6′. For entries marked with a ^, this accuracy is achieved with the lower noise rate 10^{-6} for all seven equations. Equation 2.1 is satisfied with constant $1 < c_1 < 1.1$.

Table 3: The Value of the Replication Factor r That Is the Closest Integer Solution to Equation 2.1″ for the Given Values of the Neuron Count n, the Degree d, and the Inverse Synaptic Strength k, where $k_m = 2k$ and $k_a = k$.

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 100{,}000$ neurons								
$d = 128$								
$d = 256$								
$d = 512$		2134	5170					
$d = 1024$		1000	2436					
$d = 2048$		473*	1164	2653				
$d = 4096$			562*	1284*	2809			
$d = 8192$				628*	1372*	2922		
$d = 16{,}384$					678*	1438*		
$d = 32{,}768$					339*	716*	1488*	
$d = 65{,}536$								
$n = 1{,}000{,}000$ neurons								
$d = 128$								
$d = 256$								
$d = 512$								
$d = 1024$	3571	9974	24327					
$d = 2048$		4707	11,603	26,487				
$d = 4096$		2233	5576	12,792	28,027			
$d = 8192$		1065	2692	6219	13,650	29,122		
$d = 16{,}384$			1305	3037	6688	14,265	29,899	
$d = 32{,}768$			636*	1488*	3290	7028	14,706	
$d = 65{,}536$				733	1625*	3477	7274	
$d = 131{,}072$				364*	807*	1728*	3614	7454
$d = 262{,}144$					406*	865	1806	3717*
$d = 524{,}288$								
$n = 10{,}000{,}000$ neurons								
$d = 128$								
$d = 256$								
$d = 512$								
$d = 1024$	35,693	99,248						
$d = 2048$	16,451	46,807	115,310					
$d = 4096$		22,307	55,487	126,970				
$d = 8192$		10,619	26,877	61,909				
$d = 16{,}384$		5069	13,005	30,302	66,577			
$d = 32{,}768$			6307	14,815	32,814	69,939		
$d = 65{,}536$				7258	16,155	34,635	72,347	
$d = 131{,}072$				3562	7967	17,131	35,946	
$d = 262{,}144$					3936	8487	17,838	36,887
$d = 524{,}288$							8867	18,350
$d = 1{,}048{,}576$								

Notes: Equation 2.1″ is accurate to ratio 10^{-2} (but only 10^{-1} if marked by *) and equation 2.4″ to 10^{-6}. Equations 2.2″, 2.3″, 2.5″, and 2.6″ are accurate to 10^{-6}. These accuracies are achieved even if the noise rates for equations 2.2″, 2.3″, 2.5″, and 2.6″ are 10^{-4}, 10^{-4}, 10^{-4}, and 10^{-5}, respectively.

Table 4: The Value of the Replication Factor r That Is the Closest Integer Solution to Equation 2.1″ for the Given Values of the Neuron Count n, the Degree d, and the Inverse Synaptic Strength k, Where $k_m = 2k$ and $k_a = k$.

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 100{,}000{,}000$ neurons								
$d = 128$								
$d = 256$								
$d = 512$								
$d = 1024$	356,186	991,770						
$d = 2048$	164,349	469,174	1,152,860					
$d = 4096$		222,765	555,466	1,270,184				
$d = 8192$		106,086	268,344	619,272				
$d = 16{,}384$		50,640	129,914	302,491	665,623			
$d = 32{,}768$			63,004	147,976	327,502	698,966		
$d = 65{,}536$				72,484	161,315	345,606	722,820	
$d = 131{,}072$				35,546	79,539	171,018	358,615	
$d = 262{,}144$					39,248	84,678	178,029	367,903
$d = 524{,}288$							88,413	183,035
$d = 1{,}048{,}576$								

Table 4: *Continued.*

	$k = 8$	$k = 16$	$k = 32$	$k = 64$	$k = 128$	$k = 256$	$k = 512$	$k = 1024$
$n = 1{,}000{,}000{,}000$ neurons								
$d = 128$								
$d = 256$								
$d = 512$								
$d = 1024$	3,561,948	9,917,674						
$d = 2048$	1,643,398	4,691,653	11,528,462					
$d = 4096$		2,227,722	5,554,651	12,701,863				
$d = 8192$		1,060,910	2,683,448	6,192,598				
$d = 16{,}384$		506,456	1,299,108	3,024,779	6,656,114			
$d = 32{,}768$			630,010	1,479,667	3,274,935	6,989,595		
$d = 65{,}536$				724,718	1,613,041	3,455,971	7,228,103	
$d = 131{,}072$				355,323	795,184	1,710,073	3,585,972	
$d = 262{,}144$					392,293	846,699	1,780,004	3,678,864
$d = 524{,}288$							883,950	1,830,101
$d = 1{,}048{,}576$								

Notes: Equation 2.1″ is accurate to ratio 10^{-2}, and equation 2.4 to 10^{-6}. Equations 2.2″, 2.3″, 2.5″, and 2.6″ are accurate to 10^{-6}. These accuracies are achieved even if the noise rates for equations 2.2″, 2.3″, 2.5″, and 2.6″ are 10^{-4}, 10^{-4}, 10^{-4}, and 10^{-5}, respectively.

little smaller in these tables than in Tables 1 and 2. With regard to equation 2.6, our findings, which are not detailed here, are as follows. While the parameters of Tables 1 and 2 support maximum values of $t = 1$ usually, the smaller values of r in Tables 3 and 4 (where equation 2.6 is only an approximation) lead to rather larger values of t, scattered in the range 1 to 7. Further, it turns out that instead of using $k_m = 2k_a$, as we do in these tables, we can find similar results for slightly smaller coefficients, such as $k_m = 1.95k_a$, or $k_m = 1.9k_a$, and these give even smaller values of r and larger values of t.

4 The Computational Model and Algorithms

As explained earlier, our goal is not only to show that the connectivity of the networks we consider are sufficient to provide the minimum communication bandwidth needed for realizing memorization and association, but also to show that algorithms are possible that modify the network so as to be able to execute instances of these tasks. In particular, each of these two tasks and for each representation, one needs two algorithms—one for creating the circuit, say for associating A to B in the first place, and one for subsequently executing the task, namely, causing B to fire when A fires.

For describing such algorithms, we need a model of computation. We employ the neuroidal model because it is programmable and well suited to describing algorithms (Valiant, 1994). As mentioned earlier, the neuroidal model is designed to be so simple that there is no debate that real neurons have at least as much power. It is not designed to capture all the features of real neurons.

Our algorithms are described for a variant of the neuroidal model that allows synapses to have memory in addition to weights. This has some biological support (Markram & Tsodyks, 1996b) and allows for somewhat more natural programming, even though temporary values of synaptic weights may be used instead, in principle, to simulate such states (Valiant, 1994). A brief summary of the model is as follows. A neuroidal net consists of a weighted directed graph G with a model neuron or *neuroid* at each node. A neuroid is a threshold element with some additional internal memory, which can be in one of a set of modes. The *mode* s_i of node i at an instant will specify the threshold T_i, and may also have further components such as a member q_i of a finite set of *states* Q. In particular, a mode is either firing or nonfiring and f_i has value 1 or 0 accordingly. The *weight* of an edge from node j to node i is w_{ji} and models the strength of a synapse for which j is presynaptic and i is postsynaptic. Each synapse can also have a *state* q_{ji}, which with w_{ji} forms a component of the mode s_{ji}. The only way a neuroid i can be influenced by other neuroids is through the quantity w_i which equals the sum of the weights w_{ji} over all nodes j presynaptic to i that are in firing modes. Each neuroid executes an algorithm that is local to itself and can be formally defined in terms of *mode update functions* δ and λ, for the neuroid itself and each synapse, respectively:

$$\delta(s_i, w_i) = s_i', \text{ and}$$

$$\lambda(s_i, w_i, s_{ji}, f_j) = s_{ji}'.$$

These relations express the values of the modes of the neuroid and synapses at one step in terms of the values at the previous step of the variables on which they are permitted to depend. Thus, the mode of a neuroid can depend only on its mode at the previous step and on the sum w_i of weights of synapses incoming from firing nodes. The mode of one of its synapses can depend only on the mode of the same synapse, on the mode of the neuroid, on the firing status of the presynaptic neuroid, and on the sum w_i of weights of synapses incoming from firing presynaptic neuroids. The model assumes a timing mechanism that has two components. Each transition has a period. We assume here that all transitions have a period of 1, except for *threshold transitions*, those that are triggered by $w_i \geq T_i$, which work on a faster timescale. There is a global synchronization mechanism such that, for example, if some external input is to cause the representations of two items A and B to fire simultaneously, then the nodes partaking in these representations will be caused to fire synchronously enough that the algorithms that will be caused to execute can keep in lockstep for the duration of these local algorithms. These durations will be typically no more than 10, and for the purposes of this article, just two steps.

By a disjoint representation, we mean a representation where each neuroid can represent at most one item, though one item may be represented by many neuroids. The specific disjoint representation that our algorithms support has been called a *positive representation* (Valiant, 1994). The generalization that allows a node to represent more than one item, as needed for the shared representations of the next section, we call a *positive shared representation*.

The neuroidal model allows for negative weights, which may be needed, for example, for inductive learning. However, the algorithms we describe here for JOIN and LINK do not use negative weights.

We shall start with the algorithms needed for the disjoint two-step scheme implied by relations 2.1 to 2.6. The algorithms for implementing JOIN and LINK are very similar to those that were described for the same tasks for unit weights (Valiant, 1994, algorithms 7.2 and 8.1). It is clear, however, that once graph-theoretic properties such as equations 2.1 to 2.6 are guaranteed, then a rich variety of variants of these algorithms also suffices. We shall describe these algorithms informally here.

The following algorithm for creating JOIN needs the nodes of A and B to be caused to fire at distinct time steps. The nodes that are candidates for C = A $\wedge$ B (1) are initially in "unallocated" state q1, (2) have a fixed threshold T, (3) have each synapse in initial state qq1, and (4) have all the presynaptic weights initially at the value T/k.

The algorithm acting locally on each candidate C node will act over two steps. The first step is prompted by the firing of A and the second by the firing of B one time unit later. Following these two prompts, each candidate C node initially in state q1 that has at least k connections from A and also at least k connections from B will be in state q2, indicating that it has become a C node and assigned to store something. Incoming weights from nodes other than A or B will be made zero. An incoming weight from A will equal T/x if there are $x \geq k$ of them, and those from B will equal T/y if there are $y \geq k$ of them. A candidate node that does not receive two successive prompts will return to the initial unassigned condition.

The algorithmic mechanism that realizes this outcome is the following. First, note that a node that does become a C node needs to make the incoming synapses have one of the three weight values depending on whether it comes from A or B or neither. The trick is that after the A prompt, the synapses from A will memorize the value T/x for an $x \geq k$ as its weight, and memorize the fact that it is in this transitory condition by having the synapse state have the temporary value qq2. Also, the node will memorize the fact that it is in a transitory state in which k connections from A have been found by going to state q3. At the B prompt, if at a candidate node in state q3 some $y \geq k$ synapses come from firing nodes, then these synapses can be updated to have value T/y. At the same time, the A synapses, in state qq2, can go on to take on the values T/x, and the remaining synapses the value 0. However, if no such k connections from B nodes are found at this second step, then the whole neuroid returns to its initial condition.

The reader can verify that the circuit constructed as described can execute the created conjunction using a very similar two-step process if at any later time A and B are presented at successive time instants. However, many variations are possible. For example, if the weights are set to $T/2x$ and $T/2y$ instead of T/x and T/y, then simultaneous presentation of A and B will work for recognition. Thus, one-step execution is possible even with two-step creation.

We observe here that in general, there is a chance that the set of neuroids identified to represent a conjunction are mostly previously taken, and the new ones that can be assigned form only a small fraction of r. Condition 2.2 ensures that this effect is initially limited. The situation here is akin to that of a hashing scheme in which as the memory fills up, fewer and fewer new places are available (Valiant, 1994).

We now go on to discuss an algorithm for creating LINK. We consider A to be represented in one area from which there are directed edges to an area of relay neuroids, from which in turn there are directed edges to a third area containing B. Initially, the relay neuroids have threshold T, and all weights on incoming edges weight T/k, and on outgoing edges weight 0. The relay neuroids never change, except for firing. The neuroids in B are in state q1 initially.

The algorithm for creating LINK has one step in which the representations of A and B are caused to fire at the first prompt. For the nodes in B that are in state q1 and are caused to fire, each incoming edge from a firing node is given weight T/k.

The algorithm for executing LINK is even simpler. It requires simple threshold firing at both the relay level and in the B nodes.

We now go on to discuss the shared representation expressed by relations $2.1'$ to $2.6'$. The algorithm given for creating and executing LINK in the disjoint case described above applies unchanged to the shared case also. For JOIN, the creation algorithm given has to be modified so that no distinction is made any more between allocated and unallocated nodes. Now no creation process is necessary. Execution can be realized by the following modification of the creation algorithm for the disjoint case: (1) the final state is made the same as the initial state q1, rather than a new state q2, and (2) no synaptic weights are changed at all. The evaluation algorithm is unchanged.

The descriptions of the two algorithms above assume bipartite graphs, in which A and B will be in different areas for the case of LINK and C in a different area from A and B in the case of JOIN. To adapt the algorithms to general graphs, small modifications are needed to allow for the node sets having nonzero intersections.

For the shared representation one-step algorithm, there is again no creation process. Evaluation requires again only threshold firing. For LINK, there is no difference between the one-step and two-step cases.

5 Capacity and Interference

The intention of our representations is that when all or most of the nodes representing an item fire, then the item is considered recognized. For example, the activation of sufficiently many neurons in a representation in a motor area of cortex would cause a certain muscle movement.

This style of representation gives rise to a pair of related concerns: How many items can be represented in the system? What exactly does it mean for an item to be represented if unforeseen interference from other activities in the circuit can occur?

First, we note that these concerns occur in a fundamentally novel way in our approach as compared with some previous theories. In a traditional associative memory, for example, there is just one kind of execution, retrieval, and we can assume that nothing else is going on simultaneously with an instance of it. Hence, the notion of capacity, the number of items that can be stored, is analyzable in a clean manner (Graham & Willshaw, 1997).

In this letter, we have two kinds of tasks, memorization and association. We also have diverse environments arising from different histories of past circuit creations. Further, we may want some robustness to other simultaneous activities in the circuit, and our longer-term aim may be to support further tasks. These factors make possible a large number of potential sources of interference, which we define to be the effect on the execution of an algorithm of network conditions that arise from sources not specific to that execution.

Our guarantee that the circuit acts correctly is only with respect to some specified set of noninterference conditions such as relations 2.2, 2.3, 2.5, or 2.6 and a robustness condition that, as described in the section to follow, upper-bounds the total number of nodes that are active in the whole circuit. No guarantees are implied for situations that are outside these constraints. For example, if the circuit has a "seizure" so that half of the nodes extraneous to the task at hand fire, then this is a pathological condition for which no guarantees are offered.

A second observation is that our guarantees are only probabilistic and, at least in this article, only as computed by some numerical calculations of limited precision. In particular, in order to bound the probability of error in the various relations, we have computed the tails of the Bernouilli distribution $\mathcal{B}(n, p, k)$ by adding the individual nonnegligible terms $\mathcal{T}(n, p, i)$, using double-precision calculations. Since the number of terms grows with n, our accuracy was limited to about 10^{-6} for the largest values of n that we considered.

In principle, the calculations may be performed to arbitrary accuracy in the following sense. One could compute for what minimum integer x is the probability of error less than 10^{-x} for each relation.

Doing such detailed calculations is beyond the scope of this article. It will suffice here to observe that certainly in some cases within the tables of parameters that we consider, the errors are much smaller than the claimed 10^{-6}, in fact, less than 10^{-1000} in one extremity. As an example, we give a simple analytic upper bound on the error for relation 2.2″:

$$\mathcal{B}(n, \mathcal{B}(3r/2, p, k_m), r/10) \sim 0$$

under the assumption 2.1″,

$$n\mathcal{B}(2r - r', p, k_m) \sim r$$

where, further, $k_m = 2k$. Now, for generic variables n, k, p, and b, if $k = (1 + b)np$ and $0 \le b \le 1$, then

$$\mathcal{B}(n, p, k) \le \exp(-b^2 np/3)$$

can be derived from Chernoff's bound (Angluin & Valiant, 1979), where "exp" denotes exponentiation to the base $e = 2.71\ldots$. From this bound, it follows that

$$\mathcal{B}(3r/2, p, 2k) \le \exp(-(2k/(3rp/2) - 1)^2(3rp/2)/3).$$

Now in all cases in the tables, $rp < k$ (a fact that also follows from equation 2.1″ if we assume r' to be negligible and r/n to be small enough to ensure that $k_m = 2k$ exceeds the mean). It then follows by substitution that $\mathcal{B}(3r/2, p, 2k) \le \exp(-rp/18)$. So if we choose values from the tables with $n = 10^9$, $rp \ge 180$, and $r \ge 10^7$, say, then equation 2.2″ becomes

$\mathcal{B}(10^9, \exp(-10), 10^6) \sim 0$. Applying the general bound on $\mathcal{B}$ given above a second time gives that the error in equation 2.2″ is at most $\mathcal{B}(10^9, 10^{-4}, 10^6) \leq \mathcal{B}(10^9, 10^{-4}, 2*10^5) \leq \exp(-10^5/3) \leq 10^{-1000}$. Thus, at one extremity of our range of parameters, the errors for the one relation 2.2″ are indeed extremely small.

Our point is that while for conceptually simpler models, the notion of capacity, a single value for the number of items that can be represented, makes sense, for more complex models a more appropriate way of expressing the same notion is that of upper-bounding the probability of various kind of interference in any execution of the task. For example, relation 2.2 does not give an absolute guarantee of the relevant interference's not happening. It says that if A and B have total size $3r/2$ and the graph is regarded as randomly generated with respect to those sets, then the probability of the unwanted interference is very small (e.g., 10^{-6} or 10^{-1000}). This we interpret to say, roughly, in the fixed network, that if A is fixed and of size r and B is a random set of $r/2$ nodes, then the interference effect will occur with such small probability.

6 Robustness to Noise

The discussion on capacity referred to specific interactions in the network. A more generic source of interference that can be analyzed is that due to some fixed fraction of neurons being active in the network that are extraneous to the task being executed. The main question is the fraction of extraneous neurons that can be active without interfering with the intended effects of the task at hand. We shall call that fraction the *noise rate* σ.

In general, we can refine each of our noninterference constraints to allow for the expected number $s = \sigma n$ of extraneous nodes being additionally active. We have restricted the entries in Tables 1 and 2 to those where noninterference relations 2.2, 2.3, 2.5, 2.6, 2.2′, 2.3′, and 2.6′, held even with perturbations corresponding to noise rates σ in the range from 10^{-4} to 10^{-6}. In all seven relations, we replaced the relevant quantities r or $r/2$, when they referred to the input neurons of the task, by $r+\sigma n$ or $r/2+\sigma n$, as appropriate. In particular, for the respective relations, the replacements were done for the following neuron sets: equations 2.2 and 2.2′: A, B; equations 2.3 and 2.3′: A, B, D; equation 2.5: A; and equations 2.6 and 2.6′: A_1, B. In Tables 3 and 4, the entries are restricted in an exactly analogous way. We note that with the exception of equations 2.3, 2.3′, and 2.3″, it is clear that with a lower noise rate, it is easier to satisfy these equations.

Estimates of the noise rates that can be tolerated can be made in a number of other senses also. For example, we could assume that all situations, including both circuit creations and executions, are subject to some noise rate, and solve equation 2.1 under that assumption. We also note that we have sought noise rates that can be supported by a very wide range of the parameters. Higher rates can be tolerated for individual parameter combinations.

7 Predictions

The entries in our tables are all solutions to equations 2.1, 2.1′, or 2.1″. Remarkably, there is the following simple interdependence among r, d, k, and n:

$$rd < kn < c_2 rd, \tag{$*$}$$

where for each entry in Tables 1 to 4, c_2 is a modest constant. In fact, for all entries in the tables with $k \geq 64$, it is the case that c_2 is smaller than 1.35. For entries with $k = 32$, $k = 16$, and $k = 8$, it is smaller than 1.6, 2.1, and 3.0, respectively. This relationship can be explained as follows. Equation 2.1 is of the form $\mathrm{f}(\mathrm{B}(r, p, k)) = r/n$, where f is a fixed function, here the squaring function. For const k, the expectation rp of the associated binomial distribution will stay a constant as r goes up by factor of 10 and p goes down by factor of 10, or, equivalently, as n goes up by a factor of 10 for constant d. (Note that this explains why the corresponding entries in the tables for the various values of n are approximately in the ratio of the magnitudes of n.) Since, in general, r/n is small, solutions of equation 2.1 will correspond to having combinations of r, p, k that correspond to a point somewhat above the expectation. In other words, k will be somewhat above $rp = rd/n$. Since the binomial distribution falls off exponentially above, the mean k will not be much larger than rn/d, from which we deduce that kd is a little larger than rn. It is easy to see that the same argument also holds for relation 2.1″ if $k_m = 2k$.

This simple relation can be taken as a prediction for systems that allocate memory in the style of our memorization mechanism, provided the number of representatives for a concept at the lower level, that is, A and B, is the same as at the next level, C. This is an attractive assumption for a memory system that treats all memorized concepts as "equal citizens." It may not be true for all systems. For example, in various levels of a vision or other sensory system, there may be amplification or reduction in the number of neurons that represent an item between the various levels, and in that case appropriate modifications of equations 2.1 or 2.1″ need to be solved instead.

Finally, we note that a node in our formalism may be simulating a unit that consists of more than one biological neuron. For example, the suggestion that local connections between different layers in cortex may have the effect of increasing the effective degree of a node in the long-range connection network is analyzed in detail in Valiant (1994).

8 Discussion

We have shown that networks of model neurons having the four parameters of neuron count, synapse count, strength of synapses, and switching time all within ranges widely observed in biology, can realize the two basic tasks of memory formation and association. We have given tables of values for these parameters that are consistent with the quantitative constraints that we have identified as being sufficient for the realization of these two tasks. Our positive result is that entries exist for realistic combinations of these numerical parameters. Further, the algorithms needed for creating and executing the circuits for these tasks are of the simplest kind, requiring as little as one step of vicinal or neighborly interaction.

The two basic tasks that we have considered here have been the basis for implementing a broader variety of cognitive tasks, including memorization of conjunctions and disjunctions, handling relations, and inductive learning, under the direct-action hypothesis of strong synapses (Valiant, 1994). For the less restrictive setting of the current article, any such broader implications that may follow have yet to be worked out. In particular, if items have a shared representation and these are to be the targets of inductive learning, then special challenges arise. If multiple concepts are being learned and the examples for them are intermingled in time, then having a single synapse take part in the learning of more than one concept would appear to be problematic.

This work offers apparently the first explanation of how the basic cognitive tasks that we consider here can be performed at all by neural systems that have synaptic strengths that are as weak as those that are typically observed experimentally. It is probable that different neural systems exploit different combinations of the numerical parameters, and do so in different ways. It is possible, and even probable, for example, that higher-order cognitive tasks require disjoint representations and stronger synapses. Our methodology offers a calculus for investigating such phenomena.

Acknowledgments

This work was supported in part by grants from the National Science Foundation: NSF-CCR-98-77049, NSF-CCR-03-10882, NSF-CCF-0432037, and NSF-CCF-04-0427129. I am also grateful to two anonymous referees for their helpful comments.

References

Abeles, M. (1991). *Corticonics: Neural circuits of the cerebral cortex*. Cambridge: Cambridge University Press.

Abramowitz, M., & Stegun, I. (1964). *Handbook of mathematical functions*. Cambridge, MA: National Bureau of Standards.

Ali, A. B., Deuchars, J., Pawelzik, H., & Thomson, A. M. (1998). CA1 pyramidal to basket and bistratified cell EPSPs: Dual intracellular recordings in rat hippocampal slices. *J. Physiol.*, *507*, 201–217.

Angluin, D., & Valiant, L. G. (1979). Fast probabalistic algorithms for Hamiltonian circuits and matchings. *J. Comput. and Syst. Sciences*, *8*, 155–193.

Barlow, H. B. (1972). Single units and sensation: A neuron doctrine for perceptual psychology. *Perception*, *1*, 371–394.

Beurle, R. L. (1955). Properties of a mass of cells capable of regenerating impulses. *Philos. Trans. R. Soc. Lond. [Biol.]*, *240*, 55–87.

Bollobas, B. (2001). *Random graphs*. Cambridge: Cambridge University Press.

Braitenberg, V. (1978). Cell assemblies in the cerebral cortex. In R. Heim & G. Palm (Eds.), *Theoretical approaches to complex systems* (pp. 171–188).

Braitenberg, V., & Schuz, A. (1998). *Cortex: Statistics and geometry of neuronal connectivity*. Berlin: Springer-Verlag.

Cook, S. A. (1971). The complexity of theorem proving procedures. In *Proc. 3rd ACM Symp. on Theory of Computing* (pp. 151–158). New York: ACM Press.

Feldman, J. A. (1982). Dynamic connections in neural networks. *Biol. Cybern.*, *46*, 27–39.

Feldman, J. A., & Ballard, D. H. (1982). Connectionist models and their properties. *Cog. Sci.*, *6*, 205–254.

Gerbessiotis, A. V. (2003) Random graphs in a neural computation model. *International Journal of Computer Mathematics*, *80*, 689–707.

Graham, B., & Willshaw, D. (1997). Capacity and information efficiency of the associative net. *Network: Comput. Neural Syst.*, *8*, 35–54.

Griffith, J. S. (1963). On the stability of brain-like structures. *Biophys. J.*, *3*, 299–308.

Griffith, J. S. (1971). *Mathematical neurobiology: An introduction to the mathematics of the nervous system*. New York: Academic Press.

Markram, H., & Tsodyks, M. (1996a). Redistribution of synaptic efficiency: A mechanism to generate infinite synaptic input diversity from a homogeneous population of neurons without changing the absolute synaptic efficacies. *J. Physiol. (Paris)*, *90*, 229–232.

Markram, H., & Tsodyks, M. (1996b). Redistribution of synaptic efficacy between pyramidal neurons. *Nature*, *382*, 807–810.

Papadimitriou, C. H. (1994). *Computational complexity*. Reading, MA: Addison-Wesley.

Shastri, L., & Ajjanagadde, A. (1993). From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences*, *16*(3), 417–494.

Shastri, L. (2001). A computational model of episodic memory formation in the hippocampal system. *Neurocomputing*, *38–40*: 889–897.

Thomson, A. M., Deuchars, J., & West, D. C. (1993). Large, deep layer pyramid-pyramid single axon EPSPs in slices of rat motor cortex displayed paired pulse and frequency-dependent depression, mediated presynaptically and self-facilitation, mediated postsynaptically. *J. Neurophysiol.*, *70*, 2354–2369.

Valiant, L. G. (1988). Functionality in neural nets. In *Proc. AAAI-88* (Vol. 2, pp. 629–634). San Mateo, CA: Morgan Kaufmann.

Valiant, L. G. (1994) *Circuits of the mind*. New York: Oxford University Press.

Received March 9, 2004; accepted July 6, 2004.