Although some circuits of size n require size n lg n when designed in a fault-tolerant style, almost all don't.

ON NETWORKS OF NOISY GATES

Nicholas Pippenger
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120

ABSTRACT: We show that many Boolean functions (including, in a certain sense, "almost all" Boolean functions) have the property that the number of noisy gates needed to compute them differs from the number of noiseless gates by at most a constant factor. This may be contrasted with results of von Neumann, Dobrushin and Ortyukov to the effect that (1) for every Boolean function, the number of noisy gates needed is larger by at most a logarithmic factor, and (2) for some Boolean functions, it is larger by at least a logarithmic factor.

1. Introduction

The study of networks of noisy gates was inaugurated by von Neumann [N] in 1952. One of the central results of this study (argued heuristically by von Neumann and proved rigorously by Dobrushin and Ortyukov [DO2] in 1977) is the following:

Theorem A: A function computed by a network of c noiseless gates can be computed by a network of O(c log c) noisy gates.

We shall not formulate this theorem more precisely here; a stronger result is proved as Theorem 3.1 below. Considerable effort was expended in trying to replace "O(c log c)" by "O(c)" in Theorem A, until Dobrushin and Ortyukov [DO1] proved the following:

Theorem B: Any network of noisy gates that computes the sum (mod 2) of n variables must have $\Omega(n \log n)$ gates.

In this theorem, it may be assumed that the noisy gates independently fail with some fixed probability $\varepsilon>0$ and that the network has error probability at most some fixed $\delta<1/3$. Since this function can obviously be computed by a network of O(n) noiseless gates, it follows that the replacement described above is not in general possible.

Our main results in this paper show that for a variety of functions O(c) gates suffice: for these functions the complexity in terms of noisy gates is the same, to within a constant factor, as the complexity in terms of noiseless gates. We give both specific examples and generic classes of functions (including the generic class of "almost all" functions) having this property. In view of the age of this problem, it seems to us remarkable that these are the first examples of this phenomenon in the literature.

A subsidiary goal of this paper is to give explicit (rather than "probabiliistic") constructions. On the other hand, we have made no attempt to obtain the best possible (or even "reasonable") constant factors.

2. Reliable Computation in the Presence of Noise

Our goals in this section are threefold. First, we shall discuss models for computation in the presence of noise, recalling the formulation of von Neumann [N] (which we shall call "weak" computation) and introducing a new formulation (which we shall call "strong" computation). Second, we shall discuss the transfer of results from one "basis" to another and from one "error level" to another. Third, we shall reprove the result of von Neumann to the effect that reliable computation

in the presence of noise is possible.

We shall assume that the notions of a basis (a set of Boolean functions), of a network and a formula over a basis, and of the depth and size of a network are known. If the gates of a network compute Boolean functions, then the network may be be regarded as computing a Boolean function. Similarly, if the gates of a network compute independent stochastic functions over any domain, then the network may be regarded as computing a stochastic function over that domain.

Let $B=\{0, 1\}$ be the Boolean domain. We shall say that a stochastic function $g:B^k \to B$ is the *ε-perturbation* of a function $f:B^k \to B$ if, for every $x \in B^k$, the probability that $f(x)=g(x)$ is exactly $1-\varepsilon$. We shall say that g *ε-approximates* f if, in these circumstances, this probability is at least $1-\varepsilon$.

We shall say that a network *weakly (ε, δ)-computes* a Boolean function f if, when the functions computed by the gates of the network are replaced by independent stochastic functions that are their ε-perturbations, the network computes a δ-approximation to f. This is the definition of computation by noisy networks introducted by von Neumann [N] and used by Dobrushin and Ortyukov [DO1, DO2] in the proof of Theorems A and B. For proving lower bounds (as in Theorem B), the weakest definition of computation is (other things being equal) the best. For proving upper bounds (as in Theorem A), however, this definition is open to certain objections, the gravest of which is the following. A network that weakly (ε, δ)-computes a function need not compute it in the ordinary sense. Thus, the complexity of computing a function with a noisy network may be less than the complexity of computing it with a noiseless network. (This is due simply to the fact that randomized algorithms may be used by noisy networks for weak computation.)

This objection could be met by substituting "ε-approximations" for "ε-perturbations" in the definition (and indeed the proof of Theorem B in [DO2] supports this change). We shall introduce an even stronger definition of computation, however; one that not only overcomes the above objection but also supports smooth change of basis and error level theorems. Of course, for proving upper bounds (as is our goal in this paper) the strongest definition of computation is (other things being equal) the best.

By a *regime* we shall mean a pair (A, h), where A is a set called the *domain* and $h:A \to B$ is a map called the *interpretation*. For each k, we shall define $h^k:A^k \to B^k$ by componentwise application of h. Given a regime (A, h), we shall say that a stochastic function $g:A^k \to A$ is an *ε-approximation* to a function $f:B^k \to B$ if, for every $x \in A^k$, the probability that $h(g(x))=f(h^k(x))$ is at least $1-\varepsilon$. (Note that this agrees with our earlier definition if we take A=B and h to be the identity map.)

We shall say that a network *strongly (ε, δ)-computes* a function f if, for every regime, when the functions computed by the gates of the network are replaced by independent stochastic functions over the domain of the regime that are their ε-approximations, then the network computes a δ-approximation to f.

We shall say that a basis Q is *universal* if for every Boolean function f there is a network over Q that computes f. It is well known (see, for example Muller [M3]) that if Q is finite and Q' is universal, then there exists a constant C such that, for every network over Q of size c, there is a network over Q' of size at most Cc that computes the same function. This fact greatly simplifies the complexity theory of Boolean functions, since (as long as constant factors are unimportant) theorems may be proved over any convenient finite universal basis and transfered

to other finite universal bases without additional effort. We seek to extend this principle to noisy networks so as to accomodate not only changes in the basis but also changes in the error levels.

Theorem 2.1: Let Q be a finite basis and let R be a universal basis. Then there exists a constant C such that for every network over Q of size c that strongly (ε, δ)-computes a function f, there is a network over R of size at most Cc that strongly $(\varepsilon/C, \delta)$-computes f.

Let Q be a universal basis. We shall say that $\varepsilon>0$ is *good* for Q if there exist a $\delta<1/2$ such that for every Boolean function there is a network over Q that strongly (ε, δ)-computes that function. Let $\varepsilon_0(Q)$ denote the supremum of all $\varepsilon>0$ such that ε is good for Q.

Theorem 2.2: Let Q be a finite universal basis and suppose $0<\xi<\eta<\varepsilon_0(Q)$. Then for every $\rho<1/2$ there are constants $\rho<\sigma<1/2$, C and E such that for every network over Q of size c that strongly (ξ, ρ)-computes a function f, there is a network over Q of size at most Cc+E that strongly (η, σ)-computes f.

Let Q be a universal basis and let ε be good for Q. We shall say that $\delta<1/2$ is *good* for (Q, ε) if for every Boolean function there is a network over Q that strongly (ε, δ)-computes that function. Let $\delta_0(Q, \varepsilon)$ denote the infemum of all $\delta<1/2$ such that δ is good for (Q, ε).

Theorem 2.3: Let Q be a universal basis, let $\varepsilon>0$ be good for Q and suppose $\delta_0(Q, \varepsilon)<\xi<\eta<1/2$. Then there exist constants C and E such that for every network over Q of size c that strongly (ε, η)-computes a function f, there is a network over Q of size at most Cc+E that strongly (ε, ξ)-computes f.

The proofs of these three theorems are routine and will be omitted in this preliminary version. Taken together, they allow us (as long as constant factors are unimportant) to prove theorems over any convenient basis and for any convenient error levels. We may then proceed to any universal basis Q by Theorem 2.1, to any ε good for Q by Theorem 2.2 and to any δ good for (Q, ε) by Theorem 2.3, never affecting the size by more than a constant factor.

We shall rely on Theorems 2.1-3 to justify some abuse of language in the remainder of this paper. If we say that a "noiseless network" or a network of "noiseless gates" computes a function, we shall mean that it computes that function in the ordinary sense. If we say that a "noisy network" or a network of "noisy gates" computes a function, we shall mean that it strongly (ε, δ)-computes that function for some $\varepsilon>0$ and $\delta<1/2$ that are either immaterial or specified by context. We shall speak of noisy gates "failing" with probability at most ε, and of noisy networks computing with "error probability" at most δ.

We have yet to show that any $\varepsilon>0$ is good for any finite Q. We shall do this now. In fact, we shall do more: we shall show that for every finite universal Q there is a constant $\Delta>0$ such that every $\delta>0$ is good for $(Q, \Delta\delta)$.

Theorem 2.4: For every finite universal basis Q there are constants Δ and D such that, for every $\delta>0$ and every $\varepsilon\leq\Delta\delta$, every function that can be computed by a noiseless network over Q of depth d can also be strongly (ε, δ)-computed by a noisy network over Q of depth at most Dd.

Let Q_0 denote the finite universal basis containing all 3-argument Boolean functions. We shall begin by proving Theorem 2.4 in the special case $Q=Q_0$.

Lemma 2.5: For every $0<\delta\leq 1/96$ and every $\varepsilon\leq\delta/2$, every function that can be computed by a noiseless network over Q_0 of

depth d can also be strongly (ε, δ)-computed by a noisy network over Q_0 of depth at most 2d.

Proof: The proof follows the lines of that of von Neumann [N]. We may assume that we are given a noiseless formula over Q_0 (since for every network there is a formula of the same depth that computes the same function) and construct a noisy formula over Q_0 (since every formula is a network).

We shall proceed by induction on d. For d=1 the lemma is trivial. Suppose that $d \geq 2$. Let the noiseless formula be $F=\phi(F_1, F_2, F_3)$, where ϕ is a 3-argument Boolean function and F_i, $1 \leq i \leq 3$, are noiseless formulae of depth at most d-1. By inductive hypothesis, there are noisy formulae G_i, $1 \leq i \leq 3$, of depth at most 2d-2 that compute the same functions as F_i, $1 \leq i \leq 3$, with error probability at most δ. Thus the noisy formula $G=\phi(G_1, G_2, G_3)$ has depth at most 2d-1 and computes the same function as F with error probability at most $3\delta+\varepsilon \leq 4\delta$. Let ψ denote the 3-argument majority function. The noisy formula $\psi(G, G, G)$ has depth at most 2d and computes the same function as F with error probability at most $3(4\delta)^2+\varepsilon=48\delta^2+\varepsilon$. Since $\delta \leq 1/96$ and $\varepsilon \leq \delta/2$, $48\delta^2+\varepsilon \leq \delta$. □

The proof of Theorem 2.4 from Lemma 2.5 is routine and will be ommited from this preliminary version.

Corollary 2.6: The n-argument majority function can be computed by a noisy network of size $O(n^7)$.

Proof: The carry and sum of a 1-bit full adder are 3-argument Boolean functions, and thus are computed by noiseless networks over Q_0 of depth 1. It follows that, for every k, the (2^k-1)-argument majority function can be computed by a noiseless network over Q_0 of depth 2k-3. Applying Lemma 2.5 yields a noisy network of over Q_0 of depth at most 4k-6, and therefore of size $O(3^{4k})$. Taking $k=\lceil \log_2(n+1) \rceil$ completes the proof, since $4\log_2 3 < 7$. □

3. Networks with Logarithmic Redundancy

Our goal in this section is to reprove Theorem A of von Neumann, Dobrushin and Ortyukov. Our motive for doing this is to give an explicit construction (von Neumann, Dobrushin and Ortyukov use a "probabilistic construction").

Theorem 3.1: If a Boolean function is computed by a noiseless network of size c, then it is also computed by a noisy network of size O(c log c).

For the proof we shall need a gadget that we shall call a "compressor", and a lemma giving an explicit construction of compressors. A bipartite multigraph with m inputs and m outputs, and k edges incident with each input and output, will be called an *(m, k, α, β)-compressor* if it has the following property: for every set A containing at most αm inputs, the set of outputs that are connected to at least k/2 inputs in A contains at most βm outputs.

Lemma 3.2: For every $m=p^2$ (p integral), there is an $(m, 8^{17}, 1/64, 1/512)$-compressor. Furthermore, its incidence matrix can be computed in space O(log m).

Proof: Jimbo and Maruoka [JM] show that for every such m there is bipartite multigraph with m inputs and m outputs, 8 edges incident with each input and output, and the following property: its incidence matrix is symmetric, has largest eigenvalue 8, and has second largest eigenvalue at most $5\sqrt{2}$. Furthermore, its incidence matrix can be computed in space O(log m).

If we take the 17-th power of this multigraph, we obtain a multigraph G in which every input and output is incident with $k=8^{17}$ edges and having the following property: its incidence matrix M is symmetric, has largest eigenvalue k, and second largest eigenvalue at most $j=(5\sqrt{2})^{17}$. It remains to show that G is an $(m, 8^{17}, 1/64, 1/512)$-compressor.

The largest eigenvalue of M^TM is k^2. The vector e that is 1 in each position is an eigenvector for the this eigenvalue with (e, e) = m.

Let A be a set of xm inputs with $x \leq 1/64$. Let B denote the set of outputs that are connected to at least $8^{17}/2$ inputs in A, and let B contain ym outputs. Let f be the vector that is 1 in the positions corresponding to inputs in A and 0 elsewhere. We have $(f, M^TMf) \leq k^2(f, e)^2/(e, e) + j^2(f, f) = k^2x^2m + j^2xm$. On the other hand, $(f, M^TMf) = (Mf, Mf) \geq ym(k/2)^2$, since (Mf, Mf) is the sum over all outputs of the square of the number of inputs in A adjacent to that output, and each output in B contributes at least $(k/2)^2$ to that sum. Combining these inequalities and dividing by $m(k/2)^2$ yields $y \leq 4x^2 + 4(j/k)^2x$. Since $x \leq 1/64$ and $(j/k)^2 = (25/32)^{17} < 1/64$, $y \leq 1/512$. □

Proof of Theorem 3.1: We shall assume that the noiseless network is over a basis containing only 2-argument functions. We shall construct a noisy network over a basis containing all 3-argument functions together with an 8^{17}-argument majority function (how ties are broken when exactly $8^{17}/2$ arguments are 1 is irrelevant). We shall take $\varepsilon = 1/512$, $\delta = 1/128$ and show that the noisy network strongly (ε, δ)-computes the same function as the noiseless one.

The proof follows the lines of that of Dobrushin and Ortyukov [DO2]. We shall replace each wire in the noiseless network by a "cable" of m wires (where m will be chosen later to be O(log c)), and replace each noiseless gate by a "module" containing O(m) noisy gates.

The output of the noisy network is computed by a "coda" that computes, with error probability at most $2\varepsilon = 1/256$, the majority of the m wires in the cable emerging from the module replacing the gate computing the output of the noiseless network. This coda has, by Corollary 2.6, size $O(m^7) = O((\log c)^7) = O(c \log c)$.

A wire v in a cable of the noisy network replacing a wire w in the noiseless network will be called *correct* if the interpretation of the value of v (in the current regime) is the value of w. We shall adopt a threshold $\theta = 3/512$ and say that a cable is *correct* if at least $(1-\theta)$m of its wires are correct.

It remains for us to show how to construct a module with the following property: if the cables entering it are correct, then except with probabilty at most $2(e/4)^{m/512}$, the cable emerging from it is also correct. We may then set $m = \lceil (512\log_{4/e}(512c))^{1/2} \rceil^2 = O(\log c)$. It is easy to show by induction on c that, except with probability at most $c2(e/4)^{m/512} \leq 1/256$, every cable is correct. It follows that the noisy network computes the same output as the noiseless network, except with probability $1/256+1/256 = \delta$.

The module that we construct will consist of two parts, Part A and Part B.

Part A (the "executive organ" in von Neumann's terminology) comprises m noisy gates that compute the same function as the corresponding gate in the noiseless network. In each of the two cables entering the module, at most θm wires are incorrect. If each of the m gates in Part A fails independently with probability at most ε, then except with probability at most $(e/4)^{m/512}$, at most $(2\theta+2\varepsilon)m = m/64$ wires will be incorrect in the cable emerging from Part A.

It remains for us to show how to construct Part B (the "restoring organ") with the following property: if the cable entering Part B has at most m/64 incorrect wires, then except with probability at most $(e/4)^{m/512}$, the cable emerging from Part B will be correct. Let G be an $(m, 8^{17}, 1/64, 1/512)$-compressor, as constructed in Lemma 3.1. Let each input of G

correspond to a wire entering Part B, and let each output of G correspond to a wire emerging from Part B. Consider a network of m noiseless gates, one for each output of G, each computing the majority of the 8^{17} inputs to which that output is connected in G. By the defining property of an (m, 8^{17}, 1/64, 1/512)-compressor, if at most m/64 inputs are incorrect, then at most m/512 outputs will be incorrect. If we now replace each noiseless gate by a noisy gate that fails with probability at most ε, then except with probability at most $(e/4)^{m/512}$, at most $m/512 + 2\varepsilon m = \theta m$ outputs will be incorrect, and thus the cable emerging from Part B will be correct. □

4. Functions with Bounded Redundancy

In the previous section, we saw that all Boolean functions have at most "logarithmic redundancy", in that their complexities in terms of noiseless and noisy networks differ by at most a logarithmic factor. In this section, we shall study present a variety of functions with "bounded redundancy".

For $r \geq 1$, let $s=2^r$. Let $g_r(x_0, ..., x_{r-1}, y_0, ..., y_{s-1}) = y_t$, where $t = x_0 + 2x_1 + ... + 2^{r-1}x_{r-1}$.

Theorem 4.1: For every r and $s=2^r$, g_r can be computed by a network of O(s) noisy gates.

Proof: We shall construct a network over a basis that contains all 3-argument Boolean functions. We shall take $\varepsilon=1/192$ and $\delta=1/24$ and show that the network we construct strongly (ε, δ)-computes g_r.

Consider a gate that computes the 3-argument function g_1. Clearly, g_r can be computed by a noiseless formula containing 2^r-1 such gates arranged in a tree. The gates of this formula are partitioned into levels, with the gates at the leaves in level 0 and the gate at the root at level r-2. Suppose that each gate in level k in the formula were to fail with probability at most $4\varepsilon(8\varepsilon)^k$. For any setting of the variables $x_0, ..., x_{r-1}$, the correct operation of the gates on the path from the output to the variable y_t (where $t = x_0 + 2x_1 + ... + 2^{r-1}x_{r-1}$) ensures the correct operation of the formula. Such a path contains just one gate in each level, so the failure probability of the formula would be at most $4\varepsilon(1 + 8\varepsilon + (8\varepsilon)^2 + ...) = 4\varepsilon/(1-8\varepsilon) \leq 6\varepsilon$. Our goal in the remainder of the proof is to bring about a similar state of affairs using only gates that fail with probability at most ε.

As in the proof of Theorem 3.1, we shall replace wires by cables and gates by modules, but the number of wires per cable and the number of gates per module will vary from level to level. For $1 \leq k \leq r-2$, we shall replace each wire entering a gate on level k by a cable containing 2k-1 wires and each wire leaving such a gate by a cable containing 2k+1 wires. We shall regard a cable as being correct if a majority of the wires it contains are correct.

Each module on level k will contain 2k+1 disjoint noisy networks, each computing the (2k-1)-argument majority of the wires in each of the cables entering it and applying the function g_1 to the results. Since this can be done by a noisless network of size O(k), it can be done by a noisy network of size O(k log k), for a total of $O(k^2 \log k)$ noisy gates in each module on level k. Since each noisy network has error probability at most 2ε, the module has error probability at most $2^{2k+1}(2\varepsilon)^{k+1} = 4\varepsilon(8\varepsilon)^k$, as was assumed in the calculation above.

Thus the cable emerging from the module on level r-1 is incorrect with probability at most 6ε. Appending a coda that computes the majority of the wires in this cable with error probability at most 2ε yields a network that is incorrect with probability at most $6\varepsilon+2\varepsilon=\delta$. Observing that the number of gates is O(s) completes the proof. □

Since any noiseless network that computes g_r clearly has $\Omega(2^r)$ gates, Theorem 4.1 provides a simple example of a Boolean function with bounded redundancy.

For the next theorem we shall need to consider networks with more than one output. We shall say that such a network with outputs $w_1, ..., w_m$ strongly (ε, δ)-computes $f_1, ..., f_m$ if, for every $1 \le j \le m$, the network obtained by ignoring all but the output w_j strongly (ε, δ)-computes f_j. For $a \ge 1$, let $b = 2^{2^a}$. Let $h_{a,0}(z_0, ..., z_{a-1}), ..., h_{a,b-1}(z_0, ..., z_{a-1})$ denote the b Boolean function of a Boolean arguments.

Theorem 4.2: For every a and $b = 2^{2^a}$, $h_{a,0}, ..., h_{a,b-1}$ can be computed by a network of O(b) noisy gates.

Proof: (Similar to Theorem 4.1.) □

Theorem 4.3: Any Boolean function of n Boolean arguments can be computed by a network of $O(2^n/n)$ noisy gates.

Proof: Take $a = \lfloor \log_2(n - \log_2 n) \rfloor$, $b = 2^{2^a}$, $r = n - a$ and $s = 2^r$. Take $\varepsilon = 1/192$ and $\delta = 1/12$. By Theorem 4.2 there is a network M of $O(b) = O(2^n/n)$ noisy gates that strongly $(\varepsilon, \delta/2)$-computes $h_{a,0}(z_0, ..., z_{a-1}), ..., h_{a,b-1}(z_0, ..., z_{a-1})$. By Theorem 4.1 there is a network N of $O(s) = O(2^n/n)$ noisy gates that strongly $(\varepsilon, \delta/2)$-computes $g_r(x_0, ..., x_{r-1}, y_0, ..., y_{s-1})$. It is easy to see that any Boolean function of the n Boolean arguments $x_0, ..., x_{r-1}, z_0, ..., z_{a-1}$ is strongly (ε, δ)-computed by a network of $O(2^n/n)$ noisy gates that is obtained by connecting each of the inputs $y_0, ..., y_{s-1}$ of N to one of the outputs $w_0, ..., w_{b-1}$ of M in an appropriate fashion. □

Since it is well known (see [M3]) that "almost all" Boolean functions of n Boolean arguments are computed only by noiseless networks with $\Omega(2^n/n)$ gates, Theorem 4.3 shows that "almost all" Boolean functions have bounded redundancy.

Let us say that a set of m Boolean functions $f_1(x_1, ..., x_n), ..., f_m(x_1, ..., x_n)$ is *linear* if each of the functions is the sum modulo 2 of some subset of the n Boolean arguments $x_1, ..., x_n$.

Theorem 4.4: Every set of n linear functions of n Boolean arguments can be computed by a network of $O(n^2/\log n)$ noisy gates.

Sketch of proof: Given n prescribed linear functions, we shall construct a noisy network over a basis that contains all 3-argument Boolean functions, a certain 11-argument Boolean function and a 2^{23}-argument Boolean function that computes the sum modulo 2 of its arguments. We shall take $\varepsilon = 1/2^{50}35$, $\delta = 1/2^{49}35$ and show that the noisy network we construct strongly (ε, δ)-computes the n prescribed linear functions. The network will consist of two parts, Part I and Part II.

Partition the arguments into $3n/\log_2 n$ groups, each containing $(1/3)\log_2 n$ arguments. For each group there are $n^{1/3}$ distinct linear functions of the arguments in that group. For each linear function in each group, Part I of the network will contain $2n^{1/3}$ disjoint (and therefore statistically independent) noisy networks that compute it with error probability at most δ. There are $O(n^{5/3}/\log n)$ such noisy networks, each containing $O(\log n \log\log n)$ noisy gates, so Part I has size $O(n^{5/3} \log\log n)$.

Each of the n prescribed linear functions can be computed as the sum modulo 2 of $3n/\log_2 n$ linear functions, one from each group. This can be done by a noiseless network of size $O(n^2/\log n)$. The remainder of the proof (the construction of Part II) is devoted to showing how this can also be done by a noisy network of size $O(n^2/\log n)$.

We shall use low-density parity-check codes, as introduced by Gallager [G]. Specifically, we shall use a code explicitly

constructed by Margulis [M2]. For $m=p^3-p$ (where p is a prime), this code has 2m bits, of which m are information bits and m are redundant bits. Each bit is involved in 3 parity-checks and each parity-check involves 6 bits. Finally, the code has $k \geq (\log (m/8))/(18 \log (1+\sqrt{2}))$ independent iterations.

We shall use a scheme for decoding this code due to Taylor [T1, T2]. This scheme operates on a 3-by-2m array of bits in which each column contains 3 estimates of the corresponding bit in a codeword. Each entry in the output array is a function of 11 entries in the input array. Since we assume the availability of an 11-argument gate that computes this function and fails with probability at most ε, if the entries in each row of the input array are independently incorrect with probability at most ξ, then the entries of the output array will be incorrect with probability at most $35\xi^2+\varepsilon$. The errors at the outputs will not be independent, but since the code has k independent iterations, the decoding process may be iterated k times and the foregoing inequality will be satisfied each time.

Finally, we may combine 2^{23} 3-by-2m arrays of bits to form another, by letting each entry of the output be the sum modulo 2 of the corresponding entries of the inputs. Since we assume the availability of a 2^{23}-argument gate computing the sum modulo 2 and failing with probability at most ε, if the entries of the input arrays are incorrect with probability at most ξ, then the entries of the output array will be incorrect with probability at most $2^{23}\xi+\varepsilon$.

Part II of the network will consist of $n^{2/3}$ sections, each computing $m=n^{1/3}$ of the prescribed linear functions. Each of these functions will correspond to an information bit in a codeword. This codeword is the sum modulo 2 of $3n/\log_2 n$ other codewords, one for each group. Each of the information bits in these other codewords was computed in Part I. Each of the redundant bits is a linear combination of the information bits, and therefore was also computed in Part I.

For each section and each group, we shall form a 3-by-2m array representing the appropriate codeword. The entries in different columns of each array will be computed by disjoint networks in Part I and will therefore be independent (this is why we made $2n^{1/3}$ "copies" of each subnetwork in Part I). The rows in each array need not be independent, and may be identical. Our task is to combine, for each section, the $3n/\log_2 n$ arrays corresponding to different groups into one by summing their corresponding entries modulo 2. The resulting array will contain, in any one of its rows and in the m columns corresponding to information bits, the functions to be computed by the section.

Part II will consist of a number of stages. Each stage will contain a combining layer that sums modulo 2 sets of 2^{23} arrays (thus reducing the number of arrays to be combined by a factor of 2^{23}), followed by a decoding layer. Since $23>18\log_2(1+\sqrt{2})$, the number of stages needed to combine $3n/\log_2 n$ arrays into one does not exceed the number of independent iterations. If the inputs to a stage are incorrect with probability at most δ, the outputs of the combining layer (which are the inputs to the decoding layer) will be incorrect with probability at most $2^{23}\delta+\varepsilon \leq 2^{24}\delta$. Thus the outputs of the stage will be incorrect with probability at most $35(2^{24}\delta)^2+\varepsilon \leq \delta$. Thus Part II, which has $n^{2/3}$ sections each containing $O(n^{4/3}/\log n)$ noisy gates for a total of $O(n^2/\log n)$ noisy gates, computes each of the n prescribed linear functions with error probability at most δ. □

Since it is well known that "almost all" sets of n linear functions of n Boolean arguments are computed only by noiseless networks with $\Omega(n^2/\log n)$ gates (the argument of Muller [M3] can be adapted to show this), Theorem 4.4 shows that "almost all" such sets of functions have bounded redundancy.

In contrast with this, Theorem B shows that the individual functions in such sets all have logarithmic redundancy.

5. References

[A1] R. Ahlswede, "Improvements of Winograd's Results on Computation in the Presence of Noise", *IEEE Trans. on Info. Theory,* 30 (1984) 872-877.

[A2] N. Alon, "Eigenvalues and Expanders", preprint.

[DO1] R. L. Dobrushin and S. I. Ortyukov, "Lower Bound for the Redundancy of Self-Correcting Arrangements of Unreliable Functional Elements", *Prob. of Info. Transm.,* 13 (1977) 59-65.

[DO2] R. L. Dobrushin and S. I. Ortyukov, "Upper Bound for the Redundancy of Self-Correcting Arrangements of Unreliable Functional Elements", *Prob. of Info. Transm.,* 13 (1977) 203-218.

[E] P. Elias, "Computation in the Presence of Noise", *IBM J. Res. and Devel.,* 3 (1958) 346-353.

[G] R. G. Gallager, *Low-Density Parity-Check Codes,* MIT Press, 1963.

[GG] O. Gabber and Z. Galil, "Explicit Construction of Linear-Sized Superconcentrators", *J. Comp. and Sys. Sci.,* 22 (1981) 407-420.

[JM] S. Jimbo and A. Maruoka, "Expanders Obtained from Affine Transformations", *STOC,* 17 (1985) 88-97.

[M1] G. A. Margulis, "Explicit Constructions of Concentrators", *Prob. of Info. Transm.,* 9 (1973) 325-332.

[M2] G. A. Margulis, "Explicit Constructions of Graphs without Short Cycles and Low Density Codes", *Combinatorica,* 2 (1982) 71-78.

[M3] D. E. Muller, "Complexity in Electronic Switching Circuits", *IRE Trans. on Electr. Comp.,* 5 (1956) 15-19.

[N] J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components", in C. E. Shannon and J. McCarthy (Eds.), *Automata Studies,* Princeton University Press, 1956, pp. 43-98.

[PR] W. W. Peterson and M. O. Rabin, "On Codes for Checking Logical Operations", *IBM J. Res. and Devel.,* 3 (1959) 163-168.

[T1] M. G. Taylor, "Reliable Information Storage in Memories Designed from Unreliable Components", *Bell Sys. Tech. J.,* 47 (1968) 2299-2337.

[T2] M. G. Taylor, "Reliable Computation in Computing Systems Designed from Unreliable Components", *Bell Sys. Tech. J.,* 47 (1968) 2339-2366.

[W] S. Winograd, "Coding for Logical Operations", *IBM J. Res. and Devel.,* 6 (1962) 430-436.