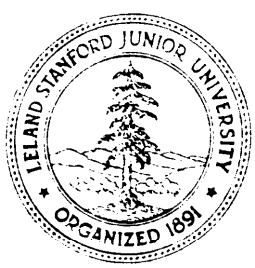
ON FAULT-TOLERANT NETWORKS FOR SORTING

by

Andrew C. Yao and F. Frances Yao

STAN-CS-79-72 1
February 1979

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY



On Fault-Tolerant Networks for Sorting*/

Andrew C. Yao and F. Frances Yao

Computer Science Department
Stanford University
Stanford, California 94305

Abstract.

The study of constructing reliable systems from unreliable components goes back to the work of von Neumann, and of Moore and Shannon. The present paper studies the use of redundancy to enhance reliability for sorting and related networks built from unreliable comparators. Two models of fault-tolerant networks are discussed. The first model patterns after the concept of error-correcting codes in information theory, and the other follows the stochastic criterion used by von Neumann and Moore-Shannon. It is shown, for example, that an additional $k(2n-3)$ comparators are sufficient to render a sorting network reliable, provided that no more than k of its comparators may be faulty.

Keywords: Batcher's network, comparators, fault-tolerant, Hamming distance, merging, networks, sorting, stochastic.

*/ This research was supported in part by National Science Foundation under grant MCS-77-05313.

1. Introduction.

Consider sorting networks that are built from comparators, where each comparator is a 2 input - 2 output device capable of sorting two numbers (Figure 1). It is of interest to construct sorting networks for n inputs using a minimum number of comparators (see Knuth [4]). The problem seems to be difficult, and so far no networks substantially better than Batcher's sorting networks (Batcher [1]) are known for general n. In this paper we look into this problem in a new setting. Suppose that some of the comparators are potentially faulty, how can we construct economic networks that still sort properly ? We shall assume that, for a faulty comparator, the inputs are directly output without a comparison (Figure 2).

The study of constructing reliable systems from unreliable components goes back to the work of von Neumann [7], and Moore and Shannon [5]. Currently, the subject of fault-tolerant computing is an active area of research (see, e.g. [6]). The present paper studies the use of redundancy to enhance reliability for a particular problem, similar in spirit to the work on switching networks by Moore and Shannon [5].

From the standpoint of analysis of algorithms, our models resemble the problem of sorting with unreliable comparisons. In that direction, a study of binary search with allowance for unreliable comparisons was done in [2].

2. Definitions and Notations.

An n-network α is a finite sequence of the form $[i_1:j_1][i_2:j_2]\ldots[i_r:j_r]$, where each pair $[i_\ell:j_\ell]$, with $1 \le i_\ell < j_\ell \le n$, is called a comparator. Any input

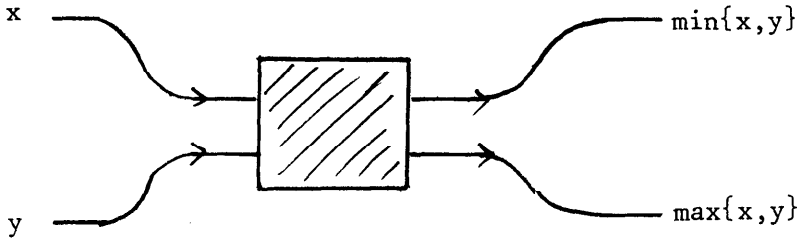


Figure 1. A comparator.

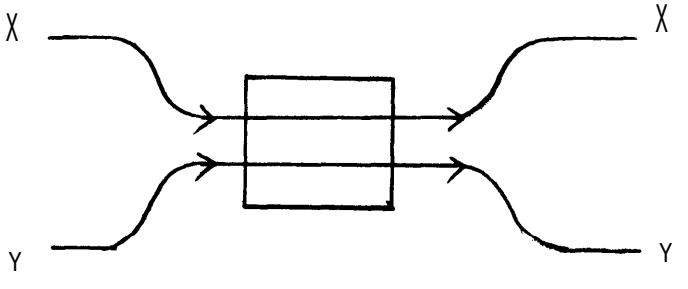


Figure 2. A faulty comparator.

vector $\vec{x} = \langle x_1, x_2, \ldots, x_n \rangle \in R^n$ of n real numbers is transformed into an output vector $\vec{y} \in R^n$ by the network α, as described below. Associate with a comparator $[i,j]$ the mapping from R^n to R^n defined by

$$\langle x_1, x_2, \ldots, x_n \rangle [i:j]_I = \langle x'_1, x'_2, \ldots, x'_n \rangle,$$

where $x'_\ell = x_\ell$ if $\ell \notin \{i,j\}$, and $x'_i = \min \{x_i, x_j\}$, $x'_j = \max \{x_i, x_j\}$. The network α then defines a mapping from R^n into R^n by successively applying the mappings induced by $[i_1:j_1], [i_2:j_2], \ldots$ and $[i_r:j_r]$. In other words, for any $\vec{x} \in R^n$, the output $\vec{y} = \vec{x}\alpha$ is defined by

$$\vec{x}^{(0)} = \vec{x},$$

$$\vec{x}^{(\ell)} = \vec{x}^{(\ell-1)} [i_\ell:j_\ell], \quad \text{for } 1 \leq \ell \leq r,$$

and $$\vec{x}\alpha = \vec{x}^{(r)}.$$

We shall represent an n-network α as shown in Figure 3, where from left to right each comparator $[i_\ell:j_\ell]$ is drawn as a vertical bar connecting the i-th and the j-th lines. We input $\vec{x} = \langle x_1, x_2, \ldots, x_n \rangle$ from the left end, with line i carrying x_i. As a comparator $[i_\ell:j_\ell]$ is passed, the smaller of the two incoming numbers moves to the upper line i_ℓ, and the larger to the lower line j_ℓ (see Figure 4 for an example). Thus, between the ℓ-th and the $(\ell+1)$-st comparators, the number carried by line i is the i-th component of the vector $\vec{x}^{(\ell)}$. In particular, $(\vec{x}\alpha)_i$ is the number found on line i at the right end of α. We call $\vec{x}^{(\ell)}$ the <u>ℓ-th state vector</u> of input $\vec{x}$ relative to α.

A vector $\vec{x} = \langle x_1, x_2, \ldots, x_n \rangle$ is <u>sorted</u> if $x_1 \leq x_2 \leq \ldots \leq x_n$. A <u>sorting network</u> for n elements, or an <u>n-sorter</u>, is an n-network a such that, for any input $\vec{x} \in R^n$, the output vector $\vec{x}\alpha$ is sorted. For instance, the network in Figure 3 is easily seen to be a 4-sorter. For each n, let S(n)

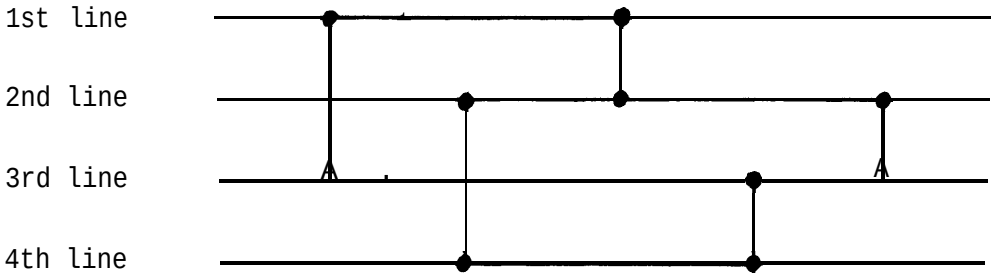


Figure 3. A 4-network a = [1:3][2:4][1:2][3:4][2:3].

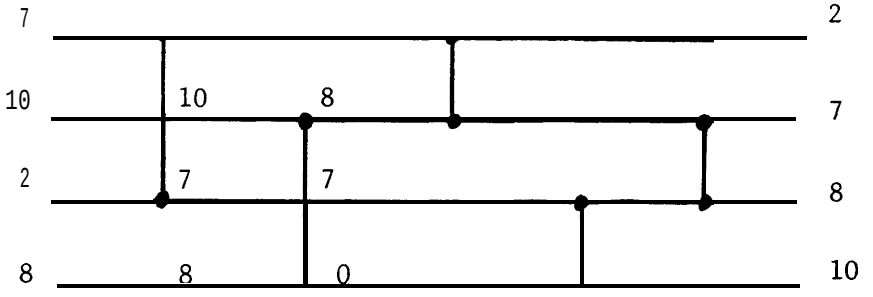


Figure 4. For input vector $\vec{x}$ = <7,10,2,8>, one has $\vec{x}^{(1)}$=<2,10,7,8>, $\vec{x}^{(2)}$= <2,8,7,10>, and output $\vec{x}\alpha = \vec{x}^{(4)}$= <2,7,8,10>.

denote the minimum number of comparators required by any n-sorter. It is known [4] that, for large n, we have $n \log_2 n \lesssim S(n) \lesssim \frac{1}{4} n(\log_2 n)^2$.

Let us now consider the situation when "faulty comparators" may be present. As the effect of having faulty comparators is equivalent to deleting them from the network, an n-sorter may no longer be an n-sorter if there are faulty comparators. Indeed, since the usual emphasis in the design of sorting networks is to avoid redundant comparisons, it is expected that every comparator is crucial in an efficient sorter. It is, therefore, an interesting question whether economic sorting networks would have to look quite different when some fault-tolerant properties are required. We shall discuss two models, with different fault-tolerant criteria, in the following sections. The first model (Section 3) patterns after the concept of error-correcting codes in information theory, and the other (Section 5) follows the criterion used in von Neumann [7] and Moore-Shannon [5] .

3. The k-Fault Model.

Let $k \geq 0$ be an integer. We are interested in constructing n-sorters which can sort properly if no more than k of its comparators are faulty. Formally, a k-tolerant n-sorter is an n-sorter α such that, if any k (or fewer) of its comparators are removed, the resulting n-network is still an n-sorter. Let $S_k(n)$ be the minimum number of comparators needed in any k-tolerant n-sorter. Trivially $S_k(n) \leq (k+1)S(n)$, since we can obtain a k-tolerant n-sorter by replacing every comparator in an optimal n-sorter with k+1 copies. Our main result in this model is the following theorem, which states that any n-sorter can be made k-tolerant by appending to it a network with $O(kn)$ comparators. The rest of this section is devoted to

a proof of Theorem 1.

Theorem 1. If α is an n-sorter, then there exists an n-network β with $k(2n-3)$ comparators, such that $\alpha\beta$ is a k-tolerant n-sorter.†

Corollary. $S_k(n) \leq S(n) + k(2n-3)$.

We need the following "zero-one principle" [4].

Lemma 1. Let ξ be an n-network. If $\vec{x}\xi$ is sorted for every $\vec{x} \in \{0,1\}^n$, then ξ is an n-sorter.

proof. See Knuth [4, Sec.5.3.4, Theorem Z]. □

Let θ denote the n-network $[1:2][2:3] \ldots [i:i+1] \ldots [n-2:n-1][n-1:n]$ $[n-2:n-1] \ldots [i:i+1] \ldots [1:2]$ (see Figure 5), and $\beta = \theta^k$ the concatenation of k such networks. Clearly, β consists of $k(2n-3)$ comparators.

Proposition 1. Let ξ be any network obtained from the n-network $\alpha\beta$ by deleting some k' comparators where $k' \leq k$. Then $\vec{x}\xi$ is sorted for any $\vec{x} \in \{0,1\}^n$.

We shall prove Proposition 1 below. Theorem 1 then follows immediately in view of Lemma 1.

Write $\xi = \alpha'\beta'$, where α' and β' are the networks resulting from α and β respectively when some a and b comparators have been removed, with $a + b \leq k$. In the remainder of this section, we will use $\vec{x}$, $\vec{y}$, etc. exclusively for vectors in $\{0,1\}^n$. For any vector $\vec{x}$, we use $\vec{x}_s$ to denote the sorted vector that has the same number of 0's as $\vec{x}$. We first show that the difference between $\vec{x}\alpha'$ and $\vec{x}_s$ is at most 2a in terms of their Hamming distance. (The Hamming distance $D(\vec{x},\vec{y})$ of $\vec{x}$ and $\vec{y}$, for $\vec{x}, \vec{y} \in \{0,1\}^n$, is the number of components where $\vec{x}$ and $\vec{y}$ differ.) We then

† We use $\alpha\beta$ to denote the concatenation of α and β.

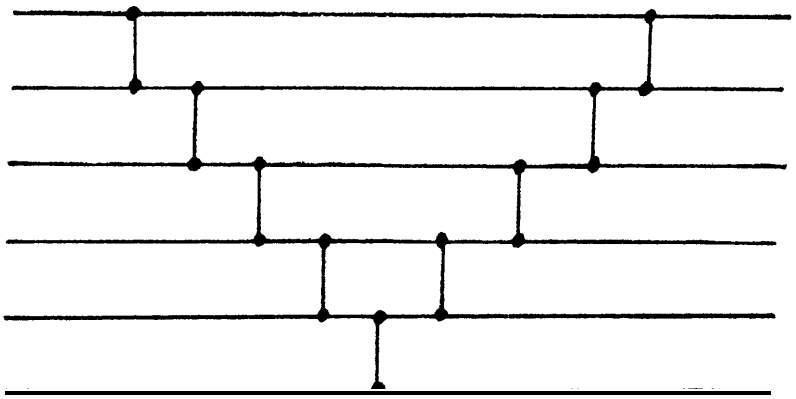


Figure 5. The network θ for six inputs.

show that the network β', with at least $k-b \geq$ a "good" copies of θ, can reduce that distance to zero.

Lemma 2. $D(\vec{x}[i:j], \vec{y}[i:j]) \leq D(\vec{x}, \vec{y})$ for any comparator $[i:j]$.

proof. It suffices to show that

$$D(\langle x_i, x_j\rangle[1:2], \langle y_i, y_j\rangle[1:2]) \leq D(\langle x_i, x_j\rangle, \langle y_i, y_j\rangle).$$

This is clearly true if the right hand side is either 0 or 2. Now, when the right hand side is 1, that means one of $\{\langle x_i, x_j\rangle, \langle y_i, y_j\rangle\}$ has exactly one 0, and the other has either two or no 0. In either case, we have $D(\vec{x}[i:j], \vec{y}[i:j]) = 1$, □

Lemma 3. $D(\vec{x}[i:j], \vec{y}) \leq D(\vec{x}, \vec{y}) + 2$.

proof. It suffices to prove that

$$D(\langle x_i, x_j\rangle[1:2], \langle y_i, y_j\rangle) \leq D(\langle x_i, x_j\rangle, \langle y_i, y_j\rangle) + 2,$$

which is obviously true. □

Lemma 4. Let α' be an n-network obtained from the n-sorter α by deleting some a comparators. Then for any $\vec{x}$,

$$D(\vec{x}\alpha', \vec{x}_s) \leq 2a .$$

where $\vec{x}_s$ is the sorted version of $\vec{x}$.

proof. Let $\vec{x}^{(\ell)}$ denote the ℓ-th state vector of $\vec{x}$ relative to α as defined in Section 2, and $\vec{y}^{(\ell)}$ the state vector of $\vec{x}$ relative to a' in the corresponding interval. Then, according to Lemmas 2 and 3,

$$D(\vec{x}^{(\ell)}, \vec{y}^{(\ell)}) \leq 2 \times (\text{the number of deleted comparators among the first } \ell \text{ of } \alpha)$$

by induction on ℓ. Therefore, $D(\vec{x}\alpha', \vec{x}\alpha) \leq 2a$, and the lemma follows since $\vec{x}\alpha = \vec{x}_s$. a

Now we consider the effect of β' on $\vec{x}\alpha'$. The network θ is designed so that if a vector $\vec{z}$ differs from $\vec{z}_s$ only by a transposition, i.e., $\vec{z} = \langle 0,0,\ldots\bar{0},\ldots 0,1,1,\ldots,\bar{1},\ldots,1,\ldots,1\rangle$ ($\bar{d}$ denotes the complement of d), then θ can carry out the desired swap for $\vec{z}$. In general, θ applied to an arbitrary vector $\vec{z}$ which is not sorted reduces the Hamming distance of $\vec{z}$ and $\vec{z}_s$ by at least 2.

Lemma 5. $D(\vec{z}\theta, \vec{z}_s) \leq D(\vec{z},\vec{z}_s) - 2$ if $D(\vec{z},\vec{z}_s) > 0$.

proof. Let $\vec{z}^{(\ell)}$ denote the state vectors of $\vec{z}$ relative to 8. Suppose there are m O's in the components of $\vec{z}$, the following facts can easily be checked.

Fact A. $D(\vec{z}^{(\ell)},\vec{z}_s) = 2\times$(the number of l's in the first m components of $\vec{z}^{(\ell)}$).

Fact B. $D(\vec{z}^{(\ell)},\vec{z}_s)$ is non-increasing as ℓ increases.

Fact C. $(\vec{z}^{(m-1)})_m = 1$.

proof of Fact C. Note that $(\vec{z}^{(m-1)})_m = \max\{z_1,z_2,\ldots,z_m\}$. Since $D(\vec{z},\vec{z}_s) > 0$, $z_1,z_2,\ldots,z_m$ can not all be 0. ❑

We now prove Lemma 5. Case(l) uppose $z_{m+1} = 0$. Then $(\vec{z}^{(m-1)})_{m+1} = 0$ and $(\vec{z}^{(m-1)})_m = 1$ by Fact C. The m-th comparator $[m:m+1]$ will swap the two components, and hence $\vec{z}^{(m)}$ has one fewer l's in the first m components than $\vec{z}^{(m-1)}$. The lemma then follows from Facts A and B. Case(2) uppose $z_{m+1} = 1$. Then Fact C implies that $(\vec{z}^{(m)})_m = 1$. It is easy to see that $(\vec{z}^{(2n-m-3)})_m = 1$ and $(\vec{z}^{(2n-m-3)})_{m+1} = 0$. The (2n-m-2)-th comparator $[m:m+1]$ then swaps these two components in $\vec{z}^{(2n-m-3)}$, causing $\vec{z}^{(2n-m-2)}$ to have one fewer l's in the first m components than $\vec{z}^{(2n-m-3)}$. The lemma again follows from Facts A and B. □

Fact D. Let γ be any n-network, then $D(\vec{z}\gamma, \vec{z}_s) \leq D(\vec{z}, \vec{z}_s)$.

Lemma 6. Assume $D(\vec{z}, \vec{z}_s) \leq 2a$, and let β' be a network obtained from β by deleting no more than k-a comparators. Then $\vec{z}\beta' = \vec{z}_s$.

proof. Write $\beta = \beta^{(1)}\beta^{(2)}\ldots\beta^{(k)}$, where each $\beta^{(i)}$ is a copy of θ. Let $\beta' = \gamma^{(1)}\gamma^{(2)}\ldots\gamma^{(k)}$ such that for some $1 \leq i_1 < i_2 < \ldots < i_a \leq k$, $\gamma^{(i_\ell)} = \beta^{(i_\ell)} = \theta$ for all ℓ. If we write $w^{(j)} = \vec{z}\gamma^{(1)}\gamma^{(2)}\ldots\gamma^{(j)}$ and $w^{(0)} = \vec{z}$, then as j increases, $D(w^{(j)}, \vec{z}_s)$ does not increase by Fact D, and in fact decreases by at least 2 when $j = i_\ell$ and $D(w^{(j)}, \vec{z}_s) \geq 0$ by Lemma 5. Thus $D(w^{(k)}, \vec{z}_s) \leq 2a - 2a = 0$. As $w^{(k)} = \vec{z}\beta'$, this implies that $\vec{z}\beta' = \vec{z}_s$. □

Proposition 1 is an immediate consequence of Lemma 4 and Lemma 6. This completes the proof of Theorem 1.

4. Networks Related to Sorting.

The k-fault model of the previous section extends naturally to comparator networks for other tasks, such as merging and selection.

An (m,n)-merging network α is an (m+n)-network such that, for any $\vec{x} \in R^{m+n}$ satisfying $x_1 \leq x_2 \leq \ldots \leq x_m$ and $x_{m+1} \leq x_{m+2} \leq \ldots \leq x_n$, the vector $\vec{x}\alpha$ is sorted. Let M(m,n) denote the minimum number of comparators needed by a. An mf-network β (minimum-finding) for n inputs is an n-network such that, for any $\vec{x} \in R^n$, $(\vec{x}\beta)_1 = \min\{x_1, x_2, \ldots, x_n\}$. Let Y(n) denote the minimum number of comparators needed by β. It is known that Y(n) = n-l and

$$\tfrac{1}{2}\, n \lg(m+1) \leq M(m,n) \leq (n+m)\left(\lceil \lg m \rceil / 2 + m/2^{\lceil \lg m \rceil} \right)$$

(Batcher [1, Sec.5.3.4], Floyd [4, Sec.5.3.4 Theorem F], Yao and Yao [8]).

The k-fault model for sorting networks can immediately be generalized to these networks. Let $M_k(m,n)$ and Y,(n) denote the corresponding minimum number of comparators for such networks with k-fault tolerance.

Theorem 1 implies immediately that

$$M_k(m,n) \leq M(m,n) + k(2(m+n) - 3).$$

For $Y_k(n)$, we have the following theorem.

Theorem 2. $Y_k(n) = (k+1)(n-1)$ for $k \geq 0$.

proof. Let α be any k-tolerant mf-network for n inputs. For each j, $1 < j \leq n$, there must be at least k+1 comparators in α of the form $[*,j]$.† Otherwise, when all comparators of the form $[*,j]$ are faulty, the input $\langle x_1, x_2, \ldots, x_n \rangle$ with $x_\ell = 1 - \delta_{\ell j}$ will not have the correct output under α. Thus, Y,(n) $\geq$ (k+l)(n-1). The reverse inequality follows from the fact that a= β^{k+1}, where $\beta = [n-1:n][n-2:n-1]\ldots[i:i+1]\ldots[1:2]$, is a k-tolerant mf-network. □

† We use $[*:j]$ to denote any comparator of the form $[i:1]$ where k=j.

5. The Stochastic-Fault Model.

In the preceding two sections, we discussed fault-tolerant networks in a framework allowing at most k faulty comparators. We have seen that the additional price paid for reliability varies with the function of the network. For sorting or merging networks, only $O(kn)$ comparators are needed in addition to the basic cost of $n\log_2 n$ or higher; whereas for minimum-finding, the extra cost is k times the original basic network.

For very large networks, the assumption of no more than k faulty comparators may be too restrictive. It is reasonable to expect that some fixed fraction, say 10^{-4}, of the basic units are faulty. A natural extension of the previous model then leads to the following question. How many comparators are needed to construct an n-sorter which remains reliable if any 10^{-4} of the comparators in it are faulty? Unfortunately, reliable networks in this case do not exist when n is large ($n > 10^4+1$). Indeed, we assert that if a fraction of $1/(n-1)$ of the comparators may be faulty, then there does not exist any reliable n-sorter in this sense. For any n-sorter α, let $j \in \{2,3,\ldots n\}$ be such that at most $1/(n-1)$ of the comparators in a are of the form $[*:j]$, then α clearly will not sort all inputs properly if all such comparators $[*:j]$ are faulty (cf. the proof of Theorem 2). In view of this fact, we will define a more relaxed, stochastic model that is very similar to the models studied in von Neumann [7], Moore and Shannon [5].

A Stochastic Model. Let $0 < \varepsilon, \delta < 1$ and n be an integer. An n-network a is an $(\&,\&)$-stochastic n-sorter if the random n-network a', obtained from α by deleting independently each comparator with any fixed probability $\delta' \leq \delta$, is an n-sorter with probability at least $1-\varepsilon$.

In an (ε,δ)-stochastic n-sorter, we shall refer to δ as the fault probability (of the comparators), and ε as the failure probability (of the network). Let $S^{(\varepsilon,\delta)}(n)$ be the minimum number of comparators required by any (ε,δ)-stochastic n-sorter. Similarly, we can define (ε,δ)-stochastic merging networks for $m + n$ inputs, (ε,δ)-stochastic mf-networks for n inputs, and the corresponding complexity $M^{(\varepsilon,\delta)}(m,n)$, $Y^{(\varepsilon,\delta)}(n)$.

A conventional method of achieving reliability is to replace a basic component by several unreliable components which simulate the basic component with high reliability [5] [7]. In our case, connecting in series m comparators, each with δ probability of fault, gives the effect of a single comparator with fault probability δ^m. If a is an n-network with N comparators (none are faulty), the network β obtained from a by replacing each comparator with m comparators in series is called the canonical m-redundant network of α. The probability for β to be a network performing the same mapping as α is at least $(1-\delta^m)^N$, which is greater than $1-\varepsilon$ for large N if $m > (\log(N/\varepsilon))/\log(1/\delta)$.

Definition. For given ε, δ and network a, the canonical m-redundant network β of α with m chosen just large enough so that β becomes an (ε,δ)-stochastic network is called the canonical (&,&)-stochastic network simulating α.

It follows from the preceding discussion that, for fixed ε, 6, an arbitrary network α with N comparators may be simulated by its canonical (ε,δ)-stochastic network which is of size $O(N\log_2 N)$. It is of interest to study the optimality of this basic strategy for enhancing reliability. As this method exploits redundancy in a primitive way, it is also not surprising that more efficient constructions exist for many problems. We shall bear out

these points in the following results. The first result illustrates the optimality of the canonical construction for minimum-finding.

Given $n > 1$ and $m > 0$, let $m_i = \lfloor (m+i-1)/(n-1) \rfloor$ for $1 \leq i < n$. The m_i's form a partition of m into $n-1$ almost equal parts in that $\sum_i m_i = m$ and $|m_i - m_j| \leq 1$ for all i, j; they are also the unique set of $n-1$ numbers satisfying these conditions (see [3, Sec.1.2.4, Ex.38]).

Define $g_{\delta,n}(m) = \prod_{1 \leq i < n} (1-\delta^{m_i})$. It is easy to see that $g_{\delta,n}(m)$ is a non-decreasing function of m for fixed n and $\delta < 1$.

Theorem 3. Let $0 < \varepsilon, \delta < 1$. Then $Y^{(\varepsilon,\delta)}(n) = m$ where m is the smallest positive integer satisfying $g_{\delta,n}(m) \geq 1 - \varepsilon$.

Corollary. For any fixed $0 < \varepsilon, \delta < 1$, $Y^{(\varepsilon,\delta)}(n) = \Theta(n\log_2 n)$ as $n \to \infty$.[†]

proof. The network $[n-1:n]^{m_1}[n-2:n-1]^{m_2} \cdots [2:3]^{m_{n-2}}[1:2]^{m_{n-1}}$ is easily seen to be a valid mf-network with probability $g_{\delta,n}(m)$, which is at least $1-\varepsilon$ by the definition of $g_{\delta,n}$. This proves that $Y^{(\varepsilon,\delta)}(n) \leq m$.

To prove the reverse inequality, we observe that, in any (ε,δ)-stochastic mf-network α for n inputs, we must have

$$\prod_{2 \leq j \leq m} (1 - \delta^{\ell_j}) \geq 1 - \varepsilon, \tag{5.1}$$

where ℓ_j is the number of comparators of the form $[*:j]$.

Fact E. Let $k > 0$ be an integer and $0 < \delta < 1$ a real number. The expression $(1-\delta^{k_1})(1-\delta^{k_2})$, where k_1 and k_2 are non-negative integers satisfying $k_1 + k_2 = k$, is maximized when $|k_1 - k_2| \leq 1$.

proof of Fact E. Otherwise, assume that the maximum is achieved at (k_1,k_2) with $k_1 > k_2 + 1$. Then

---

[†] The Θ notation means that there exist constants $a, b > 0$ such that $a(n\log_2 n) \leq Y^{(\varepsilon,\delta)}(n) \leq b(n\log_2 n)$.

$$(1 - \delta^{k_1})(1 - \delta^{k_2}) > (1 - \delta^{k_1-1})(1 - \delta^{k_2+1}).$$

This implies

$$\delta^{k_1} + \delta^{k_2} < \delta^{k_1-1} + \delta^{k_2+1},$$

or $$\delta^{k_2}(1 - \delta) < \delta^{k_1-1}(1 - \delta),$$

or $$k_2 > k_1 - 1,$$

which is a contradiction. □

In Equ.(5.1) let $\ell = \sum_{2 \le j \le n} \ell_j$. By repeated application of Fact E, the expression $\prod_{2 \le j \le n} (1-\delta^{\ell_j})$ is maximized when $|\ell_i - \ell_j| \le 1$ for all $2 \le i, j \le n$. Therefore

$$g_{\delta,n}(\ell) \ge \prod_{2 \le j \le n} (1 - \delta^{\ell_j})$$
$$\ge 1 - \varepsilon.$$

This implies that $\ell \ge m$. We have proved Theorem 3. □

To prove the corollary, let $t = \log_\delta(1 - (1-\varepsilon)^{1/(n-1)})$, $m' = \lceil t \rceil (n-1)$ and $m'' = (\lfloor t \rfloor - 1)(n-1)$. It is easy to check that $g_{\delta,n}(m') \ge 1-\varepsilon$ and $g_{\delta,n}(m'') < 1-\varepsilon$. The monotonicity of $g_{\delta,n}$ then implies that $m'' \le Y^{(\varepsilon,\delta)}(n) \le m'$. It is easy to check that, for fixed $0 < \varepsilon, \delta < 1$, we have $t = \Theta(\log n)$ as $n \to \infty$. This implies that $m' = \Theta(n \log n)$, $m'' = \Theta(n \log n)$, and hence $Y^{(\varepsilon,\delta)}(n) = \Theta(n \log n)$.

The canonical (ε, δ)-stochastic network may not always be the best solution possible, as the following example shows.

Consider the 3-sorter $\alpha = [2:3][1:2][2:3]$, and its canonical (&)-stochastic sorter $\beta = [2:3]^m[1:2]^m[2:3]^m$. By definition, the value of m is the smallest positive integer such that $(1 - 1/2^m)^3 > 1 - \varepsilon$.

It follows that

$$m = \lceil -\log_2(1 - (1-\varepsilon)^{1/3}) \rceil .$$

For $\varepsilon << 1$, the total number of comparators in β is then

$$3m \sim 3(\log_2(1/\varepsilon) + \log_2 3) + O(\varepsilon).$$

We shall now show that, there exist (&,%)-stochastic 3-sorters using only $2\log_2(1/\varepsilon) + O(\ln\ln(3/\varepsilon))$ comparators. That is, the canonical construction uses nearly 50% more comparators than is necessary when $\varepsilon \to 0$. The result follows from the next theorem.

Theorem 4. $S^{(\varepsilon,\delta)}(3) = 2\,\dfrac{\log_2(1/\varepsilon) + O(\ln\ln(3/\varepsilon))}{\log_2(1/\delta)}$

proof. We first compute $Y^{(\varepsilon,\delta)}(3)$, which according to Theorem 3 is the smallest m satisfying

$$(1 - \delta^{\lceil m/2 \rceil})(1 - \delta^{\lfloor m/2 \rfloor}) \geq 1 - \varepsilon.$$

Writing $m' = \lceil m/2 \rceil$, we obtain

$$1 - \delta^{m'} \geq (1 - \varepsilon)^{\frac{1}{2}}$$
$$= 1 - \tfrac{1}{2}\varepsilon + O(\varepsilon^2).$$

This leads to

$$m \geq 2m' - 2$$
$$\geq 2\,\frac{\log_2(1/\varepsilon) + O(1)}{\log_2(1/\delta)} .$$

As $S^{(\varepsilon,\delta)}(3) \geq Y^{(\varepsilon,\delta)}(3)$, we have proved that

$$S^{(\varepsilon,\delta)}(3) \geq 2\,\frac{\log_2(1/\varepsilon) + O(1)}{\log_2(1/\delta)} .$$

To prove the reverse inequality, we construct a 3-sorter $\alpha_\ell = [2:3]([1:2][2:3])^\ell$ (Figure 6). We shall prove that, for some constant c, the network α_ℓ with $\ell = (\log_2(1/\varepsilon) + c\,\ln\ln(3/\varepsilon))/\log_2(1/\delta)$ is an (ε,δ)-stochastic 3-sorter. This then proves the theorem.

Writing x for [2:1] and y for [1:2], we can denote α_ℓ by the string α_ℓ = yxyxy.. .xy. For added clarity, we also use the subscripted notation $\alpha_\ell = y_0 x_1 y_1 x_2 y_2 \ldots x_\ell y_\ell$ where x_i and y_i refer to the i-th [2:3] and [1:2] comparators; respectively. It is easy to see that, when comparators are deleted, the resulting network α'_ℓ fails to be a valid 3-sorter if and only if α'_ℓ does not contain a.substring which belongs to yx^+y or xy^+x, i.e., $\alpha'_\ell \in y^+x^* \cup x^*y^+ \cup x^*$. Thus the probability p_ℓ that α'_ℓ fails is less than $p_1+p_2+p_3$ where

1) $\alpha'_\ell \in y^+x^*$ with probability

$$p_1 = \sum_{1 \leq k \leq \ell+1} k \binom{\ell+1}{k} (1-\delta)^k \delta^{2\ell+1-k} \qquad (5.2)$$

since we must have $\alpha'_\ell = y_{i_1} y_{i_2} \ldots y_{i_j} x_{i_{j+1}} x_{i_{j+2}} \ldots x_{i_k}$ where $1 \leq k \leq \ell+1$, $1 \leq j \leq k$, and $0 \leq i_1 < \ldots < i_j < i_{j+1} < \ldots < i_k \leq \ell$.

After simplifications, Equ.(5.2) becomes

$$p_1 = (1-\delta)\cdot(\ell+1)\cdot\delta^\ell \sum_{1 \leq k \leq \ell+1} \binom{\ell}{k-1} (1-\delta)^{k-1} \delta^{\ell-(k-1)}$$

$$= (1-\delta)\cdot(\ell+1)\cdot\delta^\ell .$$

2) $\alpha'_\ell \in x^*y^+$ with probability $p_2 = p_1 = (1-\delta)\cdot(\ell+1)\cdot\delta^\ell$, since network α_ℓ is symmetric with respect to left-right reversal.

3) $\alpha'_\ell \in x^*$ with probability

$$p_3 = \sum_{0 \leq k \leq \ell} \binom{\ell}{k} (1-\delta)^k \delta^{2\ell+1-k}$$

$$= \delta^{\ell+1} \sum_{0 \leq k \leq \ell} \binom{\ell}{k} (1-\delta)^k \delta^{\ell-k}$$

$$= \delta^{\ell+1}.$$

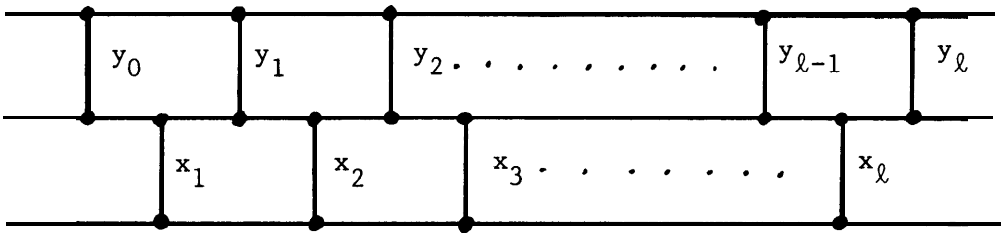


Figure 6 The network α_ℓ in the proof of Theorem 4.

Therefore $p_\ell = p_1+p_2+p_3 \leq 3(\ell+1)\delta^\ell$. It can be verified that by choosing

$$\ell = \left\lceil \frac{\ln(c/\varepsilon) + 2(\ln\ln(c/\varepsilon))}{\ln(1/\delta)} \right\rceil , \quad \text{where } \ln c \geq 3,$$

we will have $p_\ell \leq 3(\ell+1)\delta^\ell \leq \varepsilon$. This proves the theorem. □

6. Concluding Remarks.

We have studied efficient ways to achieve fault-tolerant ability in some particular problems. The canonical redundancy method sometimes yields economic networks (as for minimum-finding in both models), but not always (it works poorly for sorting in both models). It would be of great interest to find other general principles besides the canonical method.

Some related open problems:

1. For fixed ε, δ, we know that $c_1 n\log n \leq M^{(\varepsilon,\delta)}(n) \leq c_2 n(\log n)^2$. Question: Determine the order of $M^{(\varepsilon,\delta)}(n)$. Similarly, we know that $c_1 n\log n \leq S^{(\varepsilon,\delta)}(n) \leq c_2 n(\log n)^3$, and better estimates for $S^{(\varepsilon,\delta)}(n)$ are to be found. It seems that these functions should not be $O(n\log n)$, as $Y^{(\varepsilon,\delta)}(n) = \Theta(n\log n)$ and minimum-finding is intuitively a much simpler problem.

2. For fixed δ, determine $S^{(\varepsilon,\delta)}(3)$ as $\varepsilon \to 0$. In particular, is our construction optimal?

3. The interpretation of a network as a string , and the probability of fault being the probability of a random substring not containing some particular patterns gives rise to questions in a more general setting, which may be of interest by themselves.

References.

[1] K. E. Batcher, "Sorting Networks and Their Application," Proc AFIPS 1968 SJCC, Vol 32, AFIPS Press, Montvale, New Jersey, 307-314.

[2] D. J. Kleitman, A. R. Meyer, R. L. Rivest, J. Spencer and K. Winklemann, "Coping with Errors in Binary Search Procedures," Proc. 10th Ann. Symp. on Theory of Computing, San Diego, California, 1978, 227-232.

[3] D. E. Knuth, The Art of Computer Programming Vol. 1, Addison-Wesley, Reading, Mass., 2nd edition, 1973,

[4] D, E. Knuth, The Art of Computer Programming Vol. 3, Addison-Wesley, Reading, Mass., 2nd edition, 1975.

[5] E. F. Moore and C. Shannon, "Reliable Circuits Using Less Reliable Relays I-II," J. Franklin Inst., Vol 262 (1956), no.3, 191-208, no. 4, 281-297.

[6] Proceedings of the IEEE, special issue on fault-tolerant digital systems, Vol 66 (1978), no. 10, 1105-1300.

[7] J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components," Automata Studies, Princeton University Press, Princeton, New Jersey, 1956, 43-98.

[8] A. C. Yao and F. F. Yao, "Lower Bounds on Merging Networks," Journal of ACM (1976), 566-571.