A Highly Efficient Redundancy Scheme: Self-Purging Redundancy

JACQUES LOSQ, MEMBER, IEEE

Abstract—**The goals of this paper are to present an efficient redundancy scheme for highly reliable systems, to give a method to compute the exact reliability of such systems and to compare this scheme with other redundancy schemes. This redundancy scheme is self-purging redundancy, a scheme that uses a threshold voter and that purges the failed modules. Switches for self-purging systems are extremely simple: there is no replacement of the failed modules and module purging is quite simply implemented. Because of switch simplicity, exact reliability calculations are possible. The effects of switch reliability are quantitatively examined. For short mission times, switch reliability is the most important factor: self-purging systems have a probability of failure several times larger than the figure obtained when switches are assumed to be perfect. The influence of the relative frequency of the diverse types of failures (permanent, intermittent, stuck-at, multiple, ···) is also investigated. Reliability functions, mission time improvements, and switch efficiency are computed and displayed. Self-purging systems are compared with other redundant systems, like hybrid or NMR, for their relative merits in reliability gain, simplicity, cost, and confidence in the reliability estimation. The high confidence in the reliability evaluation of self-purging systems makes them a standard for the validation of several models that have been proposed to take into account switch reliability. The accuracy of the models using coverage factors can be evaluated in this way.**

Index Terms—**Convolutions, coverage factors, dormancy factors, mission time, Poisson distribution, reliability, self-purging redundancy, switch.**

I. Introduction

THE NEED for ultrareliable computers has been increasing very rapidly since the introduction of computers in areas where malfunctions can lead to catastrophes.

There are many ways to improve the reliability of digital systems. One way is to use *stand-by redundancy* [1]–[3] in which, as soon as a fault is detected, the faulty module is switched off and replaced by a spare that performs the same logic function. Another method is to use *massive redundancy* [4]–[6], in which module replication allows an instantaneous and automatic masking of the faults that occur. Combining these two methods, there are various solutions that are included under the title of *hybrid redundancy* [7]–[10] (Fig. 1).

Manuscript received July 22, 1975; revised December 8, 1975. This work was supported in part by the National Science Foundation Grant GJ-40286, and in part by the Institut de Recherche en Informatique et Automatique (IRIA), France.

The author is with the Digital Systems Laboratory, Department of Electrical Engineering and Department of Computer Science, Stanford University, Stanford, CA 94305.

Hybrid redundant systems present several advantages over both massive and stand-by redundant systems, but they require fairly complicated switching mechanisms [11], [12]. Complicated switching mechanisms introduce additional causes of system failure. Furthermore, very accurate modeling of hybrid systems that takes into account the reliability of the switching mechanisms is extremely complex. Another redundancy scheme, *self-purging redundancy* [9], [13], [14], has many of the advantages of hybrid redundancy and few of the disadvantages. Self-purging systems have very simple switching mechanisms, straighforward design, and they are simpler, cheaper, and more reliable than hybrid systems. Computation of the exact reliability of self-purging systems, including the effects of switch unreliability, is possible.

This paper focuses on the determination of the exact reliability expression for self-purging systems. The influence of the reliability of the switching mechanisms on the overall reliability will be determined. The reliability evolution as a function of time will be obtained. Simple bounds will be derived for the reliability. The results given by various models applied to self-purging systems will be compared to the exact reliability. This will characterize the degree of confidence that can be granted to each model.

II. Self-Purging Redundancy

A. Description

Single output self-purging systems are formed by a set of P modules (which are copies of the system to be upgraded by the use of redundancy), a disagreement detector, a switch, and a threshold voter. There is no differentiation between spare modules and core modules as in hybrid systems. Each module takes part in the vote-taking process as long as it is fault-free. When it fails, it is disconnected from the voter. The voter is a threshold gate [15] with a threshold of M and weight of one for each input (Fig. 2). All the modules are initially fault-free. They all send a logic 1 (or 0) to the voter which responds by sending a 1 (or 0) at its output. When an error occurs at the output of a module, this error is easily detected by comparison of the model output to the voter output. Failed modules are forced to send a 0 on their output. This is logically equivalent to disconnecting failed modules from the voter. The voter output is one if and only if the weighted sum of its in-

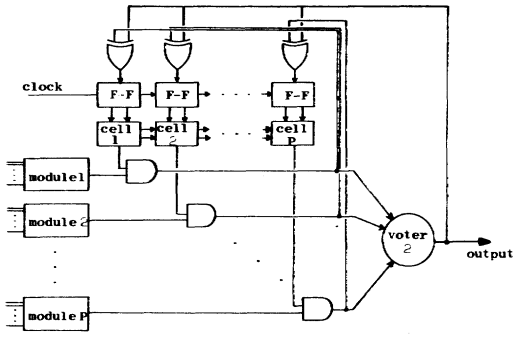

Fig. 1. Hybrid systems with TMR core, P modules, and an iterative cell switch.

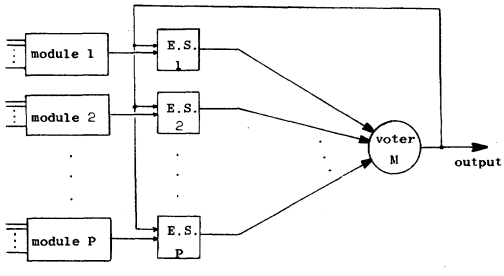

Fig. 2. Self-purging system with P modules and a voter threshold of M.

puts is equal to or greater than its threshold. So, a 0 on one of the voter inputs does not influence the voter output as long as M modules or more are correct.

Self-purging systems with a threshold of M operate properly (assuming perfect switch) as long as there are M or more fault-free modules. The interest of this redundancy scheme is that the information needed to switch off a module depends only on the state of this module. The module state is easily determined by comparison of the module output to the voter output. Another interest of self-purging systems is that modules are never switched on during system use.

Self-purging systems with multiple outputs (n outputs) can be one of two types. Type I systems have only one disagreement detector while Type II systems have n disagreement detectors (one for each output). Type II systems are more reliable because they fail only when corresponding outputs of $P - M + 1$ modules have been subject to an error, while Type I systems fail when $P - M + 1$ modules have been subject to an error.

B. Switches for Self-Purging Systems

Switches for self-purging systems can be decomposed into P *elementary switches,* one for each module, because the information needed to switch off failed modules depends only on the module state. Elementary switches are very simple. They need only detect the first disagreement between the module output and the voter output, to keep this state (has disagreed or not), and force the faulty module output to a logic 0. This can be realized with an EXCLUSIVE OR gate, a flip-flop, and an AND gate (Fig. 3).

Another advantage of self-purging systems is that it is

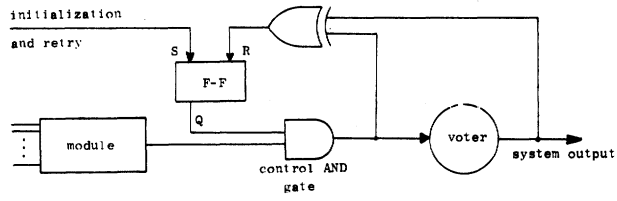

Fig. 3. Elementary switch for self-purging systems.

possible to include each elementary switch with its module to form a *modified module.* The only function of elementary switches is to force the faulty module output to a logic 0. So there is no need for elementary switches when the modules never produce erroneous 1's once they have failed. In this case, the best voter threshold, M, is one. Similarly, when failed modules never produce erroneous 0's, there is no need for switches and the best voter threshold is $P - 1$. However, in general, modules do not display this fail-safe characteristic and the voter threshold must be greater or equal to 2 and less than $P - 1$.

Self-purging systems can benefit from the use of fail-safe logic [16], [17]. When ultrahigh reliability is required, most of the system failures will be caused by switch and voter unreliability. Duplication of each elementary switch can enhance the system reliability. Each module feeds two elementary switches. The voter threshold is doubled. For short missions, the major cause of system failure is due to elementary switch failures which produce erroneous 1's on the voter inputs (module exhaustion is unlikely). So, duplication of the elementary switches reduces the probability that the voter is saturated by erroneous 1's on its inputs. However, the increase in the voter complexity (by doubling the number of its inputs) may cancel part of the reliability gain. This method is especially advantageous when the voter is realized by a ROM chip large enough to allow for twice as many inputs.

C. Retry

Many of the failures that occur in electronic components are intermittent failures. If modules in self-purging systems are declared faulty on their first disagreement with the voter, then intermittent failures will result, as hard failures do result in module removals. Clearly, this is quite expensive and even unacceptable when fast module repair cannot be provided. Retry procedures are used to distinguish between permanent and intermittent failures. Retry procedures are simply achieved for self-purging systems. To retry a module, it is sufficient to reset its elementary switch to the state "has not already failed." This can easily be done by using the asynchronous input of the flip-flop. However, it is necessary to retry a module when there are still enough fault-free modules. The easiest way is to retry a module as soon as it has disagreed with the voter for the first time.

A small counter can be included in each elementary switch to implement an automatic retry procedure. A

module will be declared faulty only if it disagrees with the voter more than once during x clock cycles. Such switches make certain that transient failures do not result in a module removal. The choice of x, the counter period, depends on the practical characteristics of the electronic components, on the environment (electromagnetic noise), and on the price one wishes to pay for such protection. However, the counter period should not be too small. Otherwise, permanent failures that produce erroneous outputs for only some of the input combinations may be mistaken for transient failures.

Self-purging systems are very advantageous to use when module repair is provided. The failed modules can be physically disconnected from their elementary switches, repaired, and reconnected without system interruption. The only necessary thing to do is to send a signal on the retry input of their elementary switches. If an open circuit corresponds to a logic 0 for the voter inputs, it is also possible to physically disconnect and reconnect elementary switches, without system interruption. So self-purging systems can be run for a very long time without system interruption if the modules can be repaired and if the repair time is shorter than the module mean-life. Such self-purging systems with repair have their reliability limited only by the voter reliability and by the multiple failure rate. A method to protect self-purging systems very efficiently from multiple failures is given in a later section.

III. Reliability of Self-Purging Systems

A. Reliability Assuming Perfect Switches

The reliability of self-purging systems with P modules and a voter with a threshold of M will be denoted by $R_{P,O,M}(T)$. In general, $R_{a,b,c}(T)$ will denote the reliability, at time T, of redundant systems with a powered modules (a modules in core), b spares, and a voter with threshold of c.

Self-purging systems with perfect voters, perfect switches, and a voter threshold of M will perform correctly as long as they have M, or more, fault-free modules. $R(T)$ denotes the reliability, at time T, of the modules. The general reliability expression for self-purging systems with perfect switches and voters is

$$R_{P,O,M}(T) = \sum_{i=M}^{P} \binom{P}{i} \cdot [R(T)]^i \cdot [1 - R(T)]^{P-i}$$

$$R_{P,O,M}(T) = 1 - \sum_{j=0}^{M-1} \binom{P}{j} \cdot [R(T)]^j \cdot [1 - R(T)]^{P-j}.$$

Self-purging redundancy is very general. When the voter threshold, M, is equal to $[P/2]$, self-purging systems are equivalent to NMR systems ($N = P$) [18], [19]. They function correctly as long as the majority of their modules are fault-free. When their threshold is 2, they correspond to hybrid systems with a TMR core and $P - 2 \cdot M + 1$ spares. In general, when their threshold is M, they correspond to hybrid systems with an NMR core ($N = 2 \cdot M - 1$) and $P - N$ spares [20].

However, self-purging systems have simpler switches than hybrid systems. The iterative cell switch developed by Siewiorek and McCluskey [11], [12], has one flip-flop and seven gates for each module (compared to one flip-flop and two gates for the elementary switches). Furthermore, the iterative cell switches for hybrid systems allow for the propagation of errors from one cell to the next one. So, the failure of one cell of the array can affect the other cells. So, switches for hybrid systems are less reliable than switches for self-purging systems. Consequently, hybrid systems are less reliable than the equivalent self-purging systems when large dormancy factors are not taken advantage of.

B. Exact Reliability Expression

The exact reliability of self-purging systems cannot be obtained by combinational methods alone. For example, let us consider a self-purging system with 5 modules and a threshold of 2. Let us assume that at time T, one module and its elementary switch (module A and switch S_A) have failed to stuck-at-one, another module (module B) is stuck-at-one, and everything else is correct. Then, at time T, the system has failed if module B has failed after both module A and switch S_A, but the system is operating correctly if the failure in B has occurred before the failure in either A or S_A. In the first case, the voter has been saturated with erroneous 1's and all the fault-free modules have been purged. As can be seen from this example, the problem exists because the switches are fed by the voter output and not by the correct module output. The *effective threshold* will denote the real threshold of the voter minus the number of voter inputs that are stuck-at-one (because of switch failures). Incorrect diagnosis by the switches happens only upon occurrence of stuck-at-one errors (erroneous 1's) when the effective threshold has been reduced to one or upon the occurrence of multiple failures which saturate the voter effective threshold. The problem of multiple failures will be treated in a following section. In this part, the failures will be considered to be Poisson distributed and the clock cycle extremely short compared to the module mean-life.

If one assumes that self-purging systems fail as soon as the effective threshold is reduced to 1, one obtains a lower bound on the reliability. If it is assumed that diagnosis is possible as long as the effective threshold is greater than zero, one obtains an upper bound. Both bounds can be obtained by combinational methods. The exact reliability is closer to the upper bound than to the lower one because, when the effective threshold is reduced to one, correct operation is possible as long as there is no stuck-at-one failure.

1) States of Modified Modules: The state of the modified module i is completely determined by the state of the line O_i, Y_i, and control AND gate A_i (Fig. 4). Any of these lines (gates) can be either fault-free or faulty. The

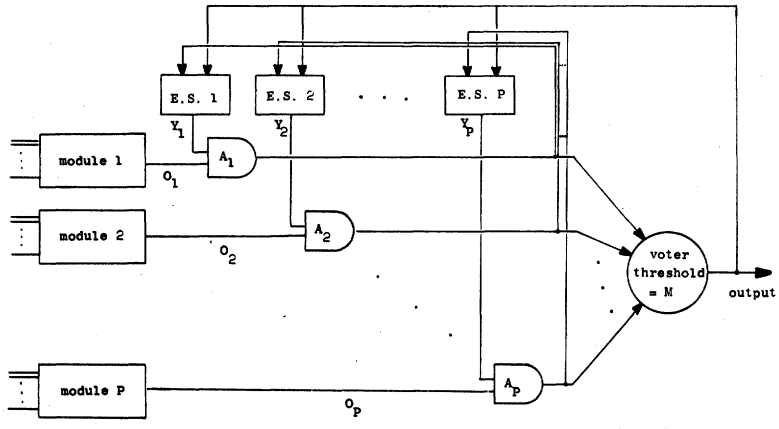

Fig. 4. General representation of self-purging systems with P modules and a voter threshold of M.

failures that affect the lines O_i and Y_i and the control AND gates A_i can be regrouped into two classes. The first class regroups the stuck-at-zero failures. The second class encompasses all other permanent failures. For simplification, the second class will be called the class of stuck-at-one failures. Any failure that can cause one erroneous logic 1 to be produced on the line O_i (Y_i, A_i) falls into the second class. This is the class of unsafe failures [21] because they can decrease the effective threshold and, consequently, endanger subsequent error detections. Stuck-at-zero failures, on the other hand, cannot reduce the effective threshold and they appear as module purges to the voter. This extended definition of stuck-at failures allows classification of every failure (bridging faults, unwanted races, ···). The effects of transient failures in the modules are normally cancelled by the automatic retry procedure. However, transient failures can cause transient system errors when the effective threshold is reduced to one. The effects of transient failures on overall reliability will be discussed in a later section. The states of the lines O_i, Y_i, and of the control AND gates A_i can be listed as good (fault-free), stuck-at-zero or stuck-at-one to mark the difference between the safe and unsafe failures.

For reliability computation, the number of states for the modified modules can be reduced from 27 (3 for each O_i, Y_i, and A_i) to 7 as follows:

- g the module and its elementary switch are fault-free (O_i, Y_i, and A_i in the fault-free state);
- s-a-g O_i and A_i in the fault-free state but Y_i in the stuck-at-one state;
- f-0 Y_i and A_i in the fault-free state, but O_i in the stuck-at-one state (module output forced to zero);
- s-a-0 O_i or Y_i in the stuck-at-zero state, but A_i in the fault-free state;
- s-a-1 O_i and Y_i in the stuck-at-one state, but A_i in the fault-free state;
- a-0 A_i in the stuck-at-zero state, whatever the states of O_i and Y_i;
- a-1 A_i in the stuck-at-one state, whatever the states of O_i and Y_i.

The probability of the modified module states can be obtained by solving a Markoff chain. The transition matrix M is fully determined by the failure rates, λ, μ, and ν, and the ratios, X_m, X_s, and X_a of stuck-at-one failures to the total number of failures in the modules, elementary switches (less the control AND gate), and the control AND gates, respectively (Table I). The state probability matrix N, Table II, gives the state probability, at time T, of the modified modules as a function of their initial state.

2) *Probability of Fault-Free Operation with Effective Threshold Larger than One:* When the effective threshold is greater than one, the fault detection and diagnosis performed by the elementary switches are perfect. Self-purging systems perform correctly as long as the number of modified modules in state g or s-a-g is greater or equal to the effective threshold M'. M' is equal to M minus the number of modified modules in state s-a-1 or a-1. Fault-free voters are also required for correct operation. In the following, voters will be ignored for they can be viewed as being in cascade with systems provided with perfect voters. However, this is a pessimistic model, for failures on the voter inputs can be viewed as modified module failures. Let $[P_1, P_2, P_3, P_4, P_5, P_6, P_7]$ be the first row of the matrix N. Then the probability, R', that self-purging systems with P modules and voter threshold of M survive until time T with an effective threshold greater than one is

$$R' = \sum_{\substack{e+g<M-1 \\ c+d+f\leq P-M}} \binom{P}{a\ b\ c\ d\ e\ f\ g} \cdot P_1^a \cdot P_2^b \cdot P_3^c \cdot P_4^d \cdot P_5^e \cdot P_6^f \cdot P_7^g.$$

R' is the lower bound of reliability. The upper bound, R_u, can be obtained similarly if we assume that correct error detection is still possible when the effective threshold is reduced to one. These bounds are very tight, except for small values of T. Their difference is equal to the probability that $M-1$ modules are stuck-at-one and that their elementary switches are also stuck-at-one (or the control AND gates).

3) *Probability of Correct Operation with an Effective*

TABLE I
Transition Matrix *M*

Present state \ Next state	g	s.a.g	f.0	s.a.0	s.a.1	a.0	a.1
g	$1-dT.(\lambda+\mu+\nu)$	$\mu.X_s.dT$	$\lambda.X_m.dT$	$[(1-X_m).\lambda+(1-X_s).\mu].dT$	0	$(1-X_a).\nu.dT$	$X_a.\nu.dT$
s.a.g	0	$1-dT.(\lambda+\nu)$	0	$(1-X_m)\lambda.dT$	$X_m.\lambda.dT$	$(1-X_a).\nu.dT$	$X_a.\nu.dT$
f.0	0	0	$1-dT.(\mu+\nu)$	$(1-X_s)\mu.dT$	$X_s.\mu.dT$	$(1-X_a).\nu.dT$	$X_a.\nu.dT$
s.a.0	0	0	0	$1-\nu.dT$	0	$(1-X_a).\nu.dT$	$X_a.\nu.dT$
s.a.1	0	0	0	0	$1-\nu.dT$	$(1-X_a).\nu.dT$	$X_a.\nu.dT$
a.0	0	0	0	0	0	1	0
a.1	0	0	0	0	0	0	1

TABLE II
State Probability Matrix N (N = limit MT/dT, $dT \to 0$)

	g	s.a.g	f.0	s.a.0	s.a.1	s.0	s.1
g	$e^{-(\lambda+\mu+\nu).T}$	$X_s.e^{-(\lambda+\nu).T}.(1-e^{-\mu.T})$	$X_m.e^{-(\mu+\nu).T}.(1-e^{-\lambda.T})$	$e^{-\nu.T}.[1-((1-X_m)e^{-\lambda T}+X_m).((1-X_s)e^{-\mu T}+X_s)]$	$e^{-\nu.T}.X_m.(1-e^{-\lambda.T}).X_s.(1-e^{-\mu.T})$	$(1-X_a).(1-e^{-\nu.T})$	$X_a.(1-e^{-\nu.T})$
s.a.g	0	$e^{-(\lambda+\nu).T}$	0	$(1-X_m).e^{-\nu.T}.(1-e^{-\lambda.T})$	$X_m.e^{-\nu.T}.(1-e^{-\lambda.T})$	$(1-X_a).(1-e^{-\nu.T})$	$X_a.(1-e^{-\nu.T})$
f.0	0	0	$e^{-(\mu+\nu).T}$	$(1-X_s).e^{-\nu.T}.(1-e^{-\mu.T})$	$X_s.e^{-\nu.T}.(1-e^{-\mu.T})$	$(1-X_a).(1-e^{-\nu.T})$	$X_a.(1-e^{-\nu.T})$
s.a.0	0	0	0	$e^{-\nu.T}$	0	$(1-X_a).(1-e^{-\nu.T})$	$X_a.(1-e^{-\nu.T})$
s.a.1	0	0	0	0	$e^{-\nu.T}$	$(1-X_a).(1-e^{-\nu.T})$	$X_a.(1-e^{-\nu.T})$
a.0	0	0	0	0	0	1	0
a.1	0	0	0	0	0	0	1

Threshold of One: When the effective threshold is reduced to one, the occurrence of stuck-at-one failures cannot be detected. However, it is possible that the effective threshold increases, going from one to two, for example, when the control AND gate of a modified module previously in state s-a-1 suffers a stuck-at-zero failure. So, the probability that a self-purging system survives for a time period u, given that the effective threshold was initially equal to one, is equal to the probability that it survives with an effective threshold always equal to one plus a convolution term giving the probability that the effective threshold increases and that the resulting system survives until the end of the period. The convolution term is complex for it can involve up to $2(P - M) + 1$ integrations (one for every change of the effective threshold).

If one assumes that the control AND gates are perfect (or fail only in the stuck-at-one mode), effective thresh-

olds are decreasing functions of time. Let $R''_{a,b,c,d,e,0,g}(u)$ be the probability that a self-purging system survives a time period u, given that it has initially a, b, c, d, e, 0, and g modified modules in the states g, s-a-g, f-0, s-a-0, s-a-1, a-0, and a-1, respectively. Then

$$R''_{a,b,c,d,e,0,g}(u) = \text{Prob(no stuck-at-one failure occurs during the time period } u) \cdot \text{Prob(at the end of the period there is at least one module in state } g \text{ or s-a-}g \mid \text{no s-a-1}).$$

These terms can be obtained from the matrix N.

4) Reliability Expression: The general expression for the reliability of self-purging systems regroups the probability of correct operation with an effective threshold greater than one plus a convolution term giving the probability of survival with an effective threshold of one.

For self-purging systems with perfect control AND gates, and a voter threshold of two, the reliability, $R_{P,0,2}(T)$, is expressed as

$$R_{P,0,2}(T') = R' + \sum \int_0^T b \cdot P_{a,b,c,d,0}(\tau) \cdot X_m \cdot \lambda \cdot R''_{a,b-1,c,d,1}(T-\tau) \cdot d\tau + \sum \int_0^T c \cdot P_{a,b,c,d,0}(\tau) \cdot X_s \cdot \mu \cdot R''_{a,b,c-1,d,1}(T-\tau) \cdot d\tau.$$

Summations are taken over all valid states with an effective threshold of two. The terms $P_{a,b,c,d,0}$ are the probabilities of the system states with a, b, c, d, 0, 0, and 0 modules, respectively, in the states g, s-a-g, f-0, s-a-0, s-a-1 a-0 and a-1. All these terms can be obtained directly from the matrix N.

IV. Performances of Self-Purging Systems

A. Reliability and Mission Time

The reliability function of a few self-purging systems is displayed in Fig. 5. All the systems have been chosen with a threshold of two because they offer the best overall performances. Systems with larger thresholds need more fault-free modules in order to function. The module complexity is expressed by the number of gates inside each module. Module stuck-at-one and stuck-at-zero failures are equilikely. The rate of stuck-at-one and stuck-at-zero failures in the elementary switches have been computed directly from the failure rate of their gates and flip-flop. Flip-flops were assumed to be composed of ten gates. Both the EXCLUSIVE OR gates and the flip-flops are equilikely to fail in the stuck-at-one as in the stuck-at-zero mode. Time has been normalized. One time unit is equal to the module mean-life.

As can be expected, the best reliability gain with respect to the irredundant systems is obtained for short mission times. Similarly, the best improvement in mission time is achieved when the reliability at the end of the mission approaches one. For example, self-purging systems with 5 modules can be operated one hundred times longer than simplex systems for an end-of-mission reliability of 0.999.

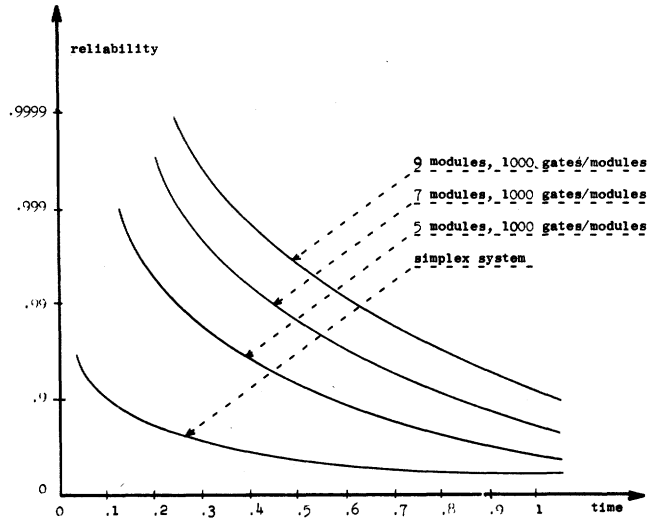

Fig. 5. Reliability of self-purging systems as a function of time.

B. Effects of Module Failure Mode

For short missions ($T \ll 1$), the reliability of self-purging systems is significantly dependent upon the preferred failure mode of the modules. For example, the probability of failure increases by 10 percent when the ratio of stuck-at-one to stuck-at-zero failures goes from 0.5 to 2. For short missions, the performances degrade when the ratio of stuck-at-one to stuck-at-zero failures increases. Module exhaustion is unlikely for short missions and the major cause of system failure is due to stuck-at-one modified modules. For large mission times, stuck-at-one failures are somewhat advantageous, but the reliability gain is negligible.

If automatic retry procedures are implemented inside the elementary switches, transient failures do not affect the system as long as the effective threshold is greater than one. When the effective threshold is reduced to one, transient module failures cause transient system errors. All the fault-free modules disagree with the voter during the duration of the transient, but they are not purged if the transient is short. The maximum increase in system probability of failure due to intermittent module failures is displayed in Fig. 6. This maximum is obtained by assuming that the probability of intermittent failures is such that the system fails as soon as the effective threshold is reduced to one. The effect of transient failures is almost insignificant when the missions are not too short. For example, the maximum increase in the probability of failure of self-purging systems with 5 modules and a threshold of 2 is less than 10 percent as long as the missions are longer than one-fifth of the module mean-life.

C. Influence of Switches on Overall Reliability

One of the most important characteristics of self-purging systems is their switch simplicity. It was shown they compare favorably with switches for hybrid sys-

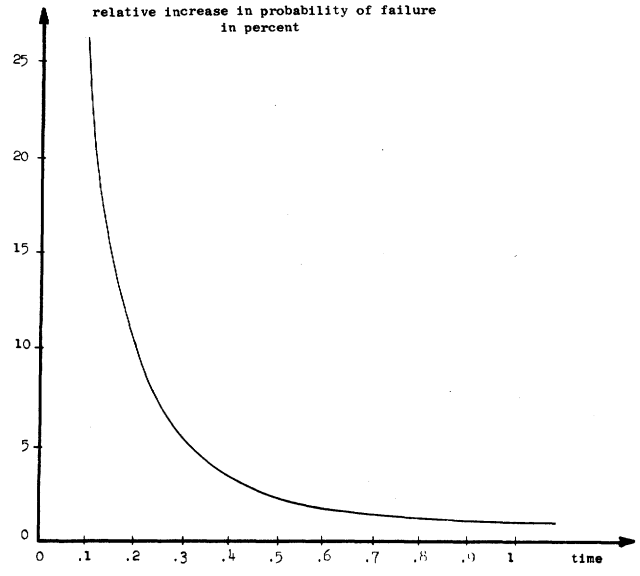

Fig. 6. Maximum increase in probability of failure of self-purging systems with 5 modules (1000 gates/module) due to transient failures.

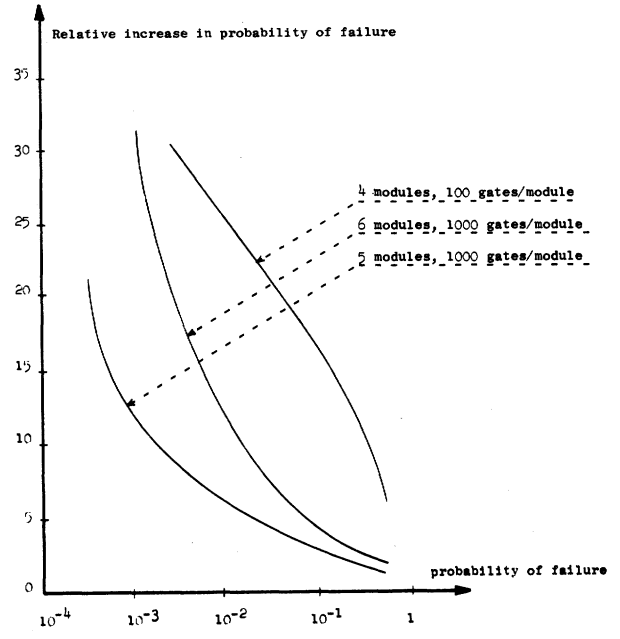

Fig. 7. Relative increase in probability of failure, in percent due to switch unreliability as a function of the system probability of failure ($X_m = X_s = 0.5$).

tems. Switch efficiency can be characterized by the relative difference between the real probability of failure and the probability of failure that is obtained when the switches are considered perfect:

$$\frac{\text{Prob(failure)} - \text{Prob(failure}|\text{the switches are perfect)}}{\text{Prob(failure}|\text{the switches are perfect)}}.$$

A ratio of zero means that switches are perfect while a ratio of x means that the real probability of failure is $(x + 1)$ times higher than what was computed assuming that switches were perfect. Fig. 7 shows this ratio for three self-purging systems. Switch efficiency approaches zero as missions get shorter. On the other hand, switch efficiency approaches one for missions of the same order or greater than the module mean-life. The importance of careful modeling for redundant systems must be emphasized even when switches are as simple as self-purging system switches. The probability of failure of self-purging systems with 6 modules is more than 50 percent larger than what is computed assuming perfect switches for mission times as long as one-fifth of the module mean-life. Also, the curves of Fig. 7 give a simple way to obtain the real reliability directly from the reliability computed with the assumption that the switches are perfect.

D. Tolerance of Self-Purging Systems to Multiple Failures

Self-purging systems with small voter threshold (two, for example) cannot recover from multiple failures that cause several modules to be stuck-at-one. Similarly, self-purging systems with large thresholds ($P - 1$ for example) cannot recover from multiple failures that cause several modules to produce erroneous zeros. The choice of the voter threshold is critical to the tolerance of self-purging systems to multiple failures. However, the voter threshold also determines how many modules are needed for correct system operation. A simple method to solve this apparent contradiction between high tolerance to multiple failures and the small number of modules required for correct operation will be presented.

Let us consider a self-purging system that has survived until time T. It has M fault-free modules left. Time T corresponds to the end of a test and purge period. The interval between two consecutive such periods is t, so that the next one will finish at time $T + t$. It will be assumed that the purging process is perfect. The problem consists of minimizing the probability that the system does not recover, at time $T + t$, from multiple failures that affect some of the M fault-free modules during the time interval $[T, T + t]$. Let $\beta(f)$ be the probability of the multiple failures that affect f of the M fault-free modules. Let $\gamma(f,i)$ be the probability of the multiple failures that affect f modules in such a way that i of the failed modules give erroneous logic 1's at time $T + t$. Let y be the probability that a failed module produces an erroneous 1 at time $T + t$. Then we obtain the probability P_d that the self-purging system will recover at time $T + t$ if its threshold is d

$$P_d = \sum_{\substack{0 \le f \le M-d \\ 0 \le i \le d \\ i \le f}} \gamma(f,i) = \sum_{\substack{0 \le f \le M-d \\ 0 \le i < d \\ i \le f}} \beta(f) \cdot \binom{f}{i} \cdot y^i \cdot (1-y)^{f-i}.$$

The best value for the threshold maximizes the value of P_d. The difference between P_d and P_{d-1} is always negative or null for d greater than or equal to $(M/2) + 1$, whatever the relative probability of the multiple failures. The difference between P_d and P_{d-1} for d equal to $(M + 1)/2$ depends only on the ratio of $\beta[(M + 1)/2]$ over $\beta[(M - 1)/2]$. For realistic values of M ($M < 10$)

the difference between P_d and P_{d-1} for d equal to $(M+1)/2$ (or $M/2$ for M even) is positive as long as the probability of the failures that affect $[M/2]$ modules is significantly smaller that the probability of the failures that affect $[M/2]-1$ modules. So, in general (y close to 0.5, M small, and probability of the multiple failures decreasing as their multiplicity increases), the best voter threshold is equal to half the number of the remaining fault-free modules.

It is quite simple to implement, inside the switches of self-purging systems, a simple mechanism that fixes the voter threshold (or rather its effective threshold) to half the number of the remaining fault-free modules (Fig. 8). For such systems, the best threshold is reduced to two when only three fault-free modules remain. So they are able to operate with only two fault-free modules. These systems are the best self-purging systems possible. They provide the best tolerance to multiple failures during all their lifetime. They can operate with only two fault-free modules. The probability of system failure due to the transients is reduced. Similarly, for short missions, the probability of system failure due to saturation of the voter by stuck-at-one modified modules is reduced. Practically, one can safely say that their reliability is extremely close to the reliability computed under the assumption that switches are perfect.

V. Self-Purging Systems Versus Other Redundant Systems

Self-purging systems are simple and efficient, and their performances can be accurately evaluated. However, they are not necessarily the best systems for every application. For each application, different kinds of redundancy should be compared for their ability to meet the requirements, their cost, simplicity, and the confidence that can be given to the results of their models.

For extremely short missions and high cost of system failure, like short manned space flights, the best kind of redundancy is masking redundancy. Stand-by systems have their reliability severely limited by the unreliability of their fault-detection and switching mechanisms [21]. The best number of spares for extremely short missions is one, and such stand-by systems are less effective than NMR systems [21]. Hybrid and self-purging systems present the same disadvantage. They need switches which severely limit their reliability. NMR systems are the best ones; there are no switches and, consequently, they function as long as the majority of their module is fault-free.

However, for most applications, the mission times are larger than a few percent of the module mean-life. For missions ranging from one-tenth to a few tenths of the module mean-life, self-purging redundancy is the best solution. For such small missions, the beneficial effect of large dormancy factors cannot be taken advantage of significantly. Self-purging systems, with their simple switches, are more reliable than hybrid systems, Fig. 9. They also can be made to tolerate more multiple failures (Fig. 8), and simple retry procedures for transient failures are available. The confidence in the models for self-purging systems is also greater than for hybrid system models. The comparison with stand-by systems is quite difficult. Stand-by system reliability is largely dependent upon fault-detection and switching mechanisms which are complex circuits and system dependent. However, the high efficiency of self-purging systems may easily overcome the potentially greater performances of stand-by systems.

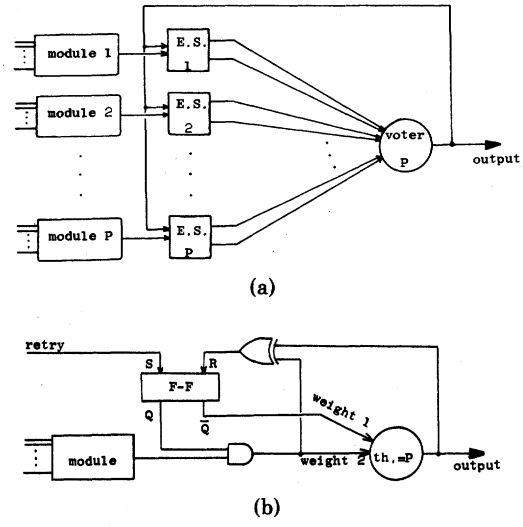

Fig. 8. (a) Self-purging systems for optimal tolerance to multiple failures. (b) Elementary switch for optimal fault tolerance.

For large mission times (T greater than 1), stand-by system performances are greater than self-purging system performances when large dormancy factors can be taken advantage of. When this is not possible, self-purging systems are the best choice.

VI. Correctness of Various Models to Take into Account Switch Reliability

Because of the accurate modeling of self-purging systems, it is possible to use them to check the accuracy of methods that are used to approximate the effect of switch unreliability on the overall reliability of redundant systems that use switching mechanisms.

The simplest and also the crudest way is to assume that fault-free switches are required for correct system operation. This model, applied to self-purging systems, gives pessimistic evaluation for the reliability (Fig. 10). Switches, especially self-purging system switches, have some inherent fault tolerance.

Another way to take into account switch reliability is to include each elementary switch in the corresponding module, compute the reliability of the modified module (defined as the cascade of the module with its elementary switch), and then compute the system reliability with the assumption that switching is perfect. This method does not take into account the interdependence between elementary switches as is the case for hybrid systems. However, this method gives good results, at least for self-purging systems, for mission times that are not too

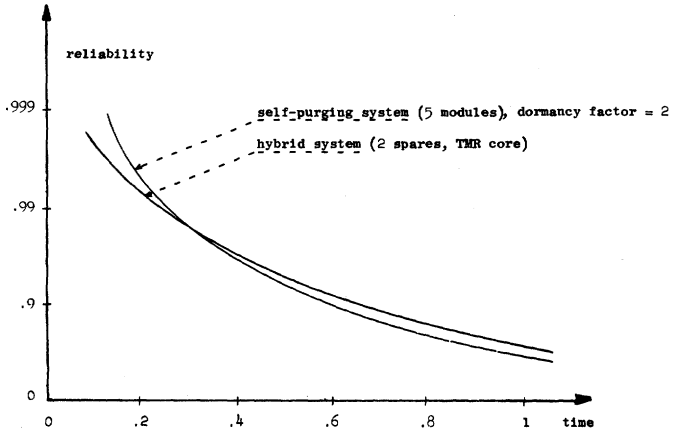

Fig. 9. Comparison between hybrid and self-purging systems (for the hybrid system, the coverage factor is equal to one minus half the ratio of switch to module complexity).

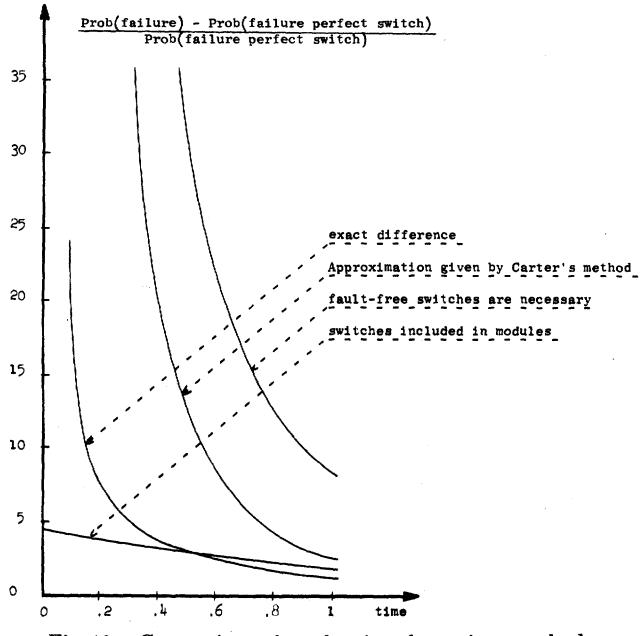

Fig. 10. Comparison of results given by various methods.

small (Fig. 10). For short missions, the results are optimistic for they do not take into account the major cause of system failure due to voter saturation by stuck-at-one modified modules still connected to the voter. So this method should not be used for ultrareliable systems because its accuracy tends to zero.

Another method has been developed by Bouricius and Carter [22]. Each replacement of a failed module by a spare is given a probability, c, of success. This probability c is called the coverage factor. Similarly, for self-purging systems, it is possible to effect a probability, c, of success for each module purge. The results given by this method are plotted in Fig. 10. The coverage factor has been chosen as the ratio of the module complexity to the complexity of the modified module. With this value, the results are pessimistic. It may be possible to obtain a better approximation by choosing a slightly higher value for c. However, it is difficult to decide, without more information, what value should be taken for the coverage factors. For hybrid systems, it is necessary to use detailed simulations or prototypes to get good approximations for parameters as critical as the coverage factors.

VII. Conclusions

Self-purging redundancy is an efficient method to increase the reliability of digital systems. Practical design is simple and straightforward. Switches can be decomposed into independent elementary switches which are only simple mechanisms to force the output of failed modules to zero. Because of their simplicity, switches are highly efficient.

Exact reliability can be computed, even though the computations are complex. Tight bounds are obtained more simply. Reliability is directly computed from the module and switch rate and modes of failures. No use of critical parameters, like coverage factors, is required. Confidence in the results is limited only by uncertainty of the failure rates.

Switch efficiency was quantitatively characterized. The importance of switch reliability increases as the requirements for system reliability become stringent. The most frequent types of module errors also influence the overall reliability. Transient failures have negligible effect on the overall reliability when switches are provided with retry procedures that are simply implemented in hardware. Self-purging systems can also be designed such that they offer the best possible tolerance to multiple failures without any decrease in their other qualities.

The major domain of application for self-purging systems regroups all applications requiring high reliability at the end of missions whose duration is of the same order as their module mean-life. Self-purging systems always offer better performances than hybrid systems when the dormancy factor is one (or is not taken advantage of). Self-purging systems designed for optimal protection against multiple failures can be used very efficiently in noisy environments.

When redundancy is used to achieve ultrahigh reliability, careful modeling must be done. Most of the methods that are used to take into account switch unreliability give only approximate results. The confidence that can be granted to their results is poor, especially for extremely short missions. On the other hand, the confidence in self-purging modeling is high and it can be used as a reference for other methods.

References

[1] A. Avizienis, "Design of fault-tolerant computers," *FJCC*, vol. 31, pp. 733–743, 1967.
[2] W. G. Bouricius, W. C. Carter, and P. R. Schneider, "Reliability modeling techniques for self-repairing computer systems," in *Proc. ACM 1969 Annu. Conf.*, pp. 295–305; also, IBM Rep. RC-2378.
[3] J. Goldberg, K. N. Levitt, and R. A. Short, "Techniques for realization of ultra-reliable spaceborn computers," Stanford Res. Inst., Menlo-Park, CA, Final Rep. Phase I, SRI project 5580, Sept. 1966.
[4] J. Von Neumann, "Probabilistic logic and the synthesis of reli-

able organisms from unreliable components," *Automata Studies (Annals of Mathematical Studies)*, C. E. Shannon and J. McCarthy, Ed. Princeton, NJ: Princeton Univ. Press, 1965, pp. 43–98.
[5] R. E. Lyions and W. Vanderkulk, "The use of triple modular redundancy to improve computer reliability," *IBM J. Res. Develop.* vol. 6, 1962, pp. 200–209.
[6] R. Toeste, "Digital circuit redundancy," *IEEE Trans. Reliability*, vol. R-13, pp. 42–61, 1964.
[7] F. P. Mathur and A. Avizienis, "Reliability analysis and architecture of a highly redundant digital system: Generalized triple modular redundancy with self-repair," *Proc. SJCC*, vol. 26, pp. 375–383, 1970.
[8] J. P. Roth, W. G. Bouricius, W. C. Carter, and P. R. Schneider, "Phase II of an architectural study for a self-repairing computer," SAMSO TR67-106, Nov. 1967.
[9] F. P. Mathur and P. T. de Sousa, "Reliability modeling and analysis of general modular redundant systems," *IEEE Trans. Reliability*, vol. R-24, pp. 296–299, Dec. 1975.
[10] J. K. Knox-Seith, "A redundancy technique for improving the reliability of digital systems," Stanford Electron. Lab., Stanford University, Stanford, CA, Tech. Rep. 4816-1, Dec. 1963.
[11] D. P. Siewiorek and E. J. McCluskey, "An iterative cell switch design for hybrid redundancy," *IEEE Trans. Comput.*, vol. C-22, pp. 290–297, Mar. 1973.
[12] ——"Switch complexity in systems with hybrid redundancy," *IEEE Trans. Comput.* vol. C-22, pp. 276–282, Mar. 1973.
[13] K. N. Chandy, C. V. Ramamoorthy, and A. Cowan, "A framework for hardware–software tradeoffs in the design of fault-tolerant computers," *FJCC, AFIPS*, pp. 55–63, 1972.
[14] W. H. Pierce, "Adaptive vote-takers improve the use of redundancy," in *Redundancy Techniques for Computing Systems*. Washington, DC: Spartan Books, 1962, pp. 229–250.
[15] R. O. Winder, "Fundamentals of threshold logic," RCA Labs, Princeton, NJ, Sci. Rep. 1, 1968.
[16] N. Tokura, T. Kasami, and A. Hashimoto, "Fail-safe logic nets," *IEEE Trans. Comput.*, vol. C-20, pp. 323–330, Mar. 1971.
[17] R. C. Ogus, "Fault-tolerance of the iterative cell array switch for hybrid redundancy," *IEEE Trans. Comput.*, vol. C-23, pp. 667–681 July 1974.
[18] J. A. Abraham and D. P. Siewiorek, "An algorithm for the accurate reliability evaluation of triple modular redundancy networks," *IEEE Trans. Comput.*, vol. C-23, pp. 682–692, July 1974.
[19] F. P. Mathur and P. T. deSousa, "Reliability models of NMR systems," *IEEE Trans. Reliability*, vol. R-24, pp. 108–113, June 1975.
[20] F. P. Mathur, "On reliability modeling and analysis of ultrareliable fault-tolerant digital systems," *IEEE Trans. Comput.*, vol. C-20, pp. 1376–1382, Nov. 1971.
[21] J. Losq, "Influence of fault-detection and switching mechanisms on the reliability of stand-by systems," in *Proc. FTC/5* (Paris, France), June 18–20, 1975, pp. 81–86.
[22] W. G. Bouricius, W. C. Carter, D. C. Jessep, R. P. Schneider, and A. B. Wadia, "Reliability modeling.for fault-tolerant computers," IEEE Trans. Comput., vol. C-20, pp. 1306–1311, 1971.

Jacques Losq (M'74) was born in Douarnenez, France, on June 1, 1949. He received the Ingénieur degree from L'Ecole Supérieure d'Electricité, Paris, France, in 1971, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, in 1972 and 1975, respectively.

He is currently working at the Digital Systems Laboratory, Stanford University, under an IBM postdoctoral fellowship. His main research topic concerns the modeling of highly reliable digital systems.

Dr. Losq is a member of ACM, Sigma Xi, and the Société Française des Ingénieurs Electriciens.

Computation-Based Reliability Analysis

JOHN F. MEYER, SENIOR MEMBER, IEEE

***Abstract*—A reliability analysis method for computing systems is considered in which the underlying criteria for "success" are based on the computations the system must perform in the use environment. Beginning with a general model of a "computer with faults," intermediate concepts of a "tolerance relation" and an "environment space" are introduced which account for the computational needs of the user and the probabilistic nature of the use environment. These concepts are then incorporated to obtain a precisely defined class of computation-based reliability measures. Formulation of a particular measure is illustrated and results, applying this measure, are compared with those of a typical structure-based analysis.**

***Index Terms*—Error tolerance, fault tolerance, reliability analysis, reliability modeling.**

Manuscript received July 15, 1975; revised December 5, 1975. This work was supported by the NASA Langley Research Center under Grant NGR 23-005-622.

The author is with the Department of Electrical and Computer Engineering (Program in Computer, Information, and Control Engineering) and the Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI 48109.

I. Introduction

THIS investigation is concerned with analytic methods of assessing the reliability of computing systems. In general, such methods are based on formal models which, at some desired level of abstraction, represent the structure of the systems to be analyzed (see [1]–[4], for example). Given a particular class of models, the ability to "rely on" a system is then quantified via one or more "reliability measures" (defined on the model class), where the formal meaning of "rely on" is usually expressed in terms of some underlying concept of system "success."

In the discussion that follows, we wish to consider analysis methods that are "computation-based" in the sense that the underlying concept of system "success" is based on the computations the system must perform (in the use environment). This is in contrast to the more usual "structure-based" analysis methods wherein suc-