NASA
Technical
Paper
2346

September 1984

NASA-TP-2346 19840026127

Validation of a Fault-Tolerant Clock Synchronization System

Ricky W. Butler
and Sally C. Johnson

NASA

NASA
Technical
Paper
2346

1984

Validation of a Fault-Tolerant Clock Synchronization System

Ricky W. Butler
and Sally C. Johnson

Langley Research Center
Hampton, Virginia

NASA
National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

Summary

The high reliability requirement of flight-crucial systems demands the use of a rigorous methodology for validation. In the 10^{-9} probability-of-failure regime, every potential area of failure must be considered. Traditional validation methods are inadequate, because they either require exorbitant lengths of time for testing or assume failure independency.

This paper presents a validation methodology for a fault-tolerant clock synchronization system utilizing formal design verification and experimental testing. The validation method relies on the formal proof process to uncover design and coding errors, and utilizes experimentation to validate the assumptions of the design proof. The experimental method is presented and described in detail. To demonstrate the feasibility of the method, the clock synchronization algorithm for the Software Implemented Fault Tolerance (SIFT) system was implemented and validated in the Langley Avionics Integration Research Laboratory (AIRLAB).

The design proof of the SIFT clock synchronization algorithm defines the maximum skew between any two clocks in the system in terms of theoretical upper bounds on certain system parameters. These upper bounds are estimated as extremely large quantiles, so large that the probability of exceeding them is less than 10^{-9}. The quantile to which each parameter must be estimated is determined by a combinatorial analysis of the system reliability. The parameters are measured by direct and indirect means, and upper bounds are estimated. A nonparametric method based on an asymptotic property of the tail of a distribution is used to estimate the upper bound of a critical system parameter. Finally, trade-offs between performance and reliability are discussed.

Introduction

Clock synchronization is an essential function in fault-tolerant multicomputer systems. Most fault-tolerant flight-control systems utilize exact-match voting algorithms that depend critically upon the synchronization of the redundant computing elements. In fact, in many systems the entire communication mechanism depends fundamentally on maintaining adequate synchronization between the replicated system clocks. Typically, a maximum clock skew is assumed and utilized as shown in figure 1. If the clock synchronization scheme fails, then system failure quickly follows. Clearly a fault-tolerant system is only as reliable as its synchronization subsystem. Therefore, any validation effort must include a careful analysis of the synchronization subsystem of a fault-tolerant computer system.

The problem of validating the fault-tolerant clock synchronization algorithm used in the Software Implemented Fault Tolerance (SIFT) computer system, an experimental fault-tolerant computer system designed for flight-crucial applications, is discussed in this report. The weaknesses of classical validation methods are discussed, and a new method of validation that relies on a combination of formal design proof and experimental testing is introduced.

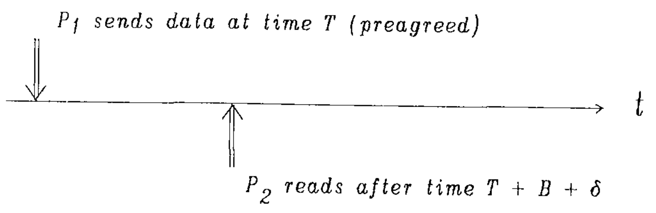
P_1 sends data at time T (preagreed)
t
P_2 reads after time $T + B + \delta$

Figure 1. Interprocess communication.

We are grateful to Larry Lee for many helpful discussions on statistical theory and for his recommendation of the Weissman estimation method.

Symbols

A_{qp}	actual skew between clock q and clock p
B	maximum broadcast time
$C(t)$	clock value at real time t
$C^{(i)}$	function defining clock during ith synchronization interval
$E(\ )$	statistical expectation of a random variable
e_{qp}	difference between actual value of clock q and value read by processor p (i.e., read error)
G_s	Gini statistic
k	number of largest observations used in Weissman's statistical method
m	maximum number of faulty processors accommodated
N	number of processors in system
p_h	probability of one or more hardware faults on a specific processor during a mission
p_{sys}	probability of system failure during a mission
p_ϵ	probability of obtaining a read error $> \epsilon$ during a single clock read
p_1	probability of estimate of maximum drift rate being too small
p_2	probability of one or more read errors $> \epsilon$ occurring on a specific processor during a mission
R	synchronization period

$R^{(i)}$	interval $[T^{(i)}, T^{(i+1)}]$
$r(T)$	real time when clock value is T
S	execution time of synchronization task
T	clock time
$T^{(i)}$	clock time at beginning of ith synchronization period
t	real time
v	a system design parameter approximately equal to mean communication delay
W_s	standardized form of Gini statistic
X_{qp}	communication delay when sending clock data from processor q to processor p
Δ_{qp}	skew between processor p's clock and processor q's clock as perceived by processor p (i.e., actual skew + read error)
δ	maximum clock skew
ϵ	maximum error in reading another processor's clock
μ	difference between v and $E(X_{qp})$, equal to $E(e_{qp})$
ρ	maximum drift rate between any two clocks
σ_ρ	standard deviation of measurements of ρ
Ω	maximum correction factor
$\hat{}$	estimate of a parameter

Inadequacy of Classical Validation Methods

Because of the criticality of synchronization systems, it is imperative that credible methods of validation be developed for these systems. However, severe requirements for flight-crucial systems, such as a probability of failure not to exceed 10^{-9} for a 10-hour flight, preclude the use of classical life testing as an assessment method. Furthermore, typical alternatives to life testing, such as combinatorial arguments or Markov models, are inadequate because they assume failure independency. Although they are physically separated, clock failures are not independent, because each clock uses values from the other clocks in the system to remain synchronized with them. Because of this failure dependency, a fault-tolerant clock synchronization algorithm is used to prevent the propagation of a clock failure to another clock in the system. The validation process must establish the correctness of this algorithm in a system context. It must be demonstrated that a single faulty clock cannot compromise the system reliability. Thus, the following failure modes must be considered:

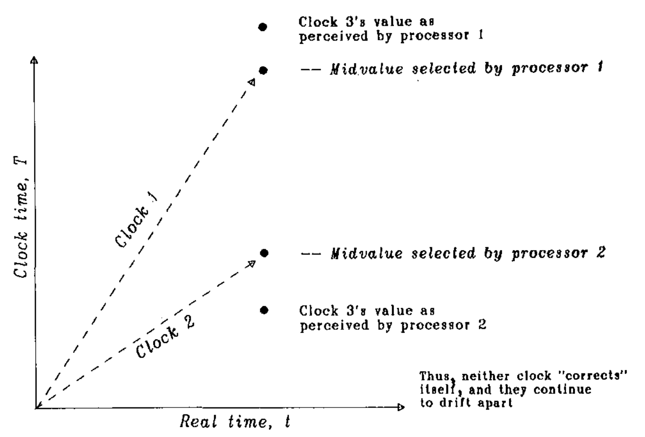
Clock 3's value as perceived by processor 1
— Midvalue selected by processor 1
Clock time, T
Clock 1
— Midvalue selected by processor 2
Clock 3's value as perceived by processor 2
Clock 2
Thus, neither clock "corrects" itself, and they continue to drift apart
Real time, t

Figure 2. Impact of malicious clock on midvalue select algorithms.

1. A majority of clocks fail before time T.
2. An error exists in the system design.
3. An error occurred in coding the synchronization algorithm.
4. Even though none of the above have occurred, the assumptions of the design have been exceeded.

Combinatorial calculations can help only in estimating the probability of failure mode 1. Yet, there is often a far greater danger of system failure due to the other failure modes. Consider the classical 3-clock midvalue select algorithm. Although "intuitively" correct, this algorithm was proven to be faulty by SRI International. (See ref. 1.) A malicious clock which sends different values to different clocks can defeat the algorithm. (See fig. 2.) To avoid the possibility of failure mode 2, SRI International developed a new algorithm and a mathematical proof characterizing the performance of this algorithm in terms of certain system parameters. A mechanical verification of the proof (i.e., using formal software verification methods and automatic theorem provers), if performed, could virtually eliminate this failure mode. The use of code verification could also eliminate failure mode 3. However, the possibility of a mode-4 failure must be considered. Fortunately, the formal proof process encapsulates precisely the design assumptions in the form of a set of axioms. Thus, the following validation method is appropriate:

1. Mathematically prove a theorem which characterizes the maximum clock skew permitted by the synchronization algorithm in terms of measurable system parameters. These parameters are defined through formal axioms.
2. Mechanically verify that the implementation code correctly implements the algorithm.

3. Experimentally verify the axioms required in the design proof above.

Although the SIFT synchronization code has not yet been mechanically checked, a mathematical design proof has been performed on the algorithm. The mechanical proof will be performed by SRI International under NASA contract NAS1-17067 during 1984 and 1985. In the following section, this algorithm and its proof are discussed.

SRI Clock Synchronization Algorithm

To discuss the SRI clock synchronization algorithm properly, it is necessary to introduce a few definitions and some notation. The theory in this section was developed by SRI under the SIFT development contract NAS1-15428 (see ref. 1).

It is convenient to define a clock as a function from real time t to clock time T: $C(t) = T$. Real time is distinguished from clock time by the use of small letters for the former and capital letters for the latter. It is sometimes useful to use the inverse clock function $r(T) = C^{-1}(T) = t$. Using this inverse function, the concept of synchronization can be defined as follows:

Definition: Two clocks r_p and r_q are synchronized to within δ of each other at time T if

$$|r_p(T) - r_q(T)| < \delta$$

Next, a good clock is defined as follows:

Definition: A clock r is a good clock during the interval $[T_1, T_2]$ if it is a monotonic, differentiable function on $[T_1, T_2]$ and if there exists a ρ such that for all T in $[T_1, T_2]$:

$$\left| \frac{dr}{dT}(T) - 1 \right| < \frac{\rho}{2}$$

Thus, the drift rate of a good clock from real time is bounded by $\rho/2$ as illustrated in figure 3.

A clock synchronization algorithm periodically resets the clocks in the system. This process may be viewed as redefining the mathematical clock function:

$$r^*(T) = r(T - A)$$

or equivalently

$$C^*(t) = C(t) + A$$

Here, the new clock C^* is obtained by incrementing clock C by A seconds. As the processors synchronize clocks every R seconds, the time base of each processor is a sequence of redefined clock functions. Using $T^{(i)}$ as the clock time at the beginning of the ith interval,

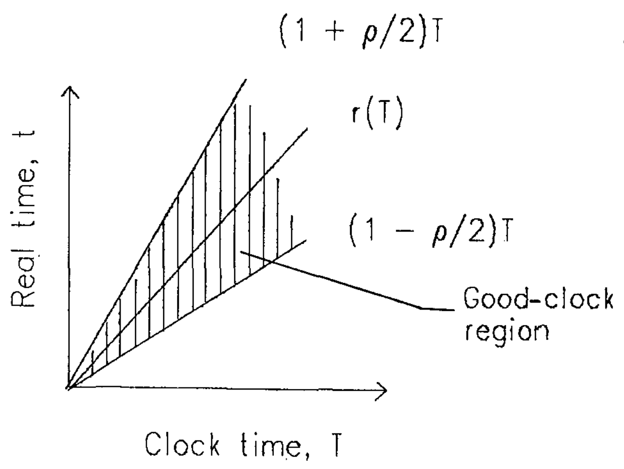

Figure 3. Definition of a good clock.

$T^{(i)} = T^{(0)} + iR$, and $R^{(i)} = [T^{(i)}, T^{(i+1)}]$. For each such interval there is a new clock definition as follows:

$$C^{(i+1)}(t) = C^{(i)}(t) + A^{(i)}$$

where $A^{(i)}$ is the ith clock correction.

The clock synchronization algorithm requires that each processor exchange clock values with every other processor during the subinterval

$$S^{(i)} = \left[T^{(i+1)} - S, T^{(i+1)}\right]$$

which is the last S seconds of the interval $R^{(i)}$. Since this clock value exchange is subject to error, it is necessary to introduce a notation and an axiom which characterize this error:

Axiom: If processors p and q are nonfaulty and their clocks are synchronized up to time $T^{(i+1)}$, then p obtains a value Δ_{qp} during the interval $S^{(i)}$, such that

$$\left| r_p^{(i)}(T_0 + \Delta_{qp}) - r_q^{(i)}(T_0) \right| < \epsilon$$

for some time T_0 in $S^{(i)}$. Thus, the error in reading another processor's clock is bounded by ϵ.

The SIFT synchronization algorithm is as follows:

Algorithm: For all clocks p,

$$C_p^{(i+1)} = C_p^{(i)} + \Delta_p$$

where

$$\Delta_p = (1/N) \sum_{r=1}^{N} \bar{\Delta}_{rp}$$

If $r \neq p$ and $|\Delta_{rp}| < \Omega$, then

$$\bar{\Delta}_{rp} = \Delta_{rp}$$

else

$$\bar{\Delta}_{rp} = 0$$

where

$$\Omega \approx \delta + \epsilon$$

The following theorem was proved by Leslie Lamport and Michael Melliar-Smith of SRI International:

Theorem: If

$3m < N$,

$$\delta \geq \frac{N}{N-3m}\left\{2\epsilon + \rho\left[R + 2\left(\frac{N-m}{N}\right)S\right]\right\},$$

$\delta \geq \delta_o + \rho R$,

$\delta \ll R$,

and

$\delta \ll \epsilon/\rho$

and if no more than m processes are faulty up to time $T^{(i+1)}$, then for all clocks p and q:

1. If processes p and q are nonfaulty up to time $T^{(i+1)}$, then for all values of T in $R^{(i)}$:

$$\left|r_p^{(i)}(T) - r_q^{(i)}(T)\right| < \delta$$

2. If process p is nonfaulty up to time $T^{(i+1)}$, then

$$\left|r_p^{(i+1)}(T) - r_p^{(i)}(T)\right| < \Omega$$

where

ϵ	maximum error in reading another processor's clock
ρ	maximum drift rate between any two clocks
m	maximum number of faulty clocks accommodated
N	number of clocks in system
R	synchronization period
S	execution time of synchronization task
Δ_{qp}	skew between processor p's clock and processor q's clock as perceived by processor p (i.e., actual skew + read error)

Assumptions of Design Proof

The design proof effectively establishes the correctness of the algorithm, assuming a set of axioms is correct. Many of these axioms are well-established mathematical theorems. Other axioms define the behavior of the computer system on which the algorithm executes. For example, the SRI design proof assumes that every processor can read another processor's clock to within an error of ϵ. The correctness of such assumptions must be established by experimentation. The following is a list of the system behavior axioms which are assumed in the SRI design proof:

1. The maximum drift rate between any two working clocks is $< \rho$.
2. If two processors are nonfaulty, then one processor can read another processor's clock to within an error of ϵ.
3. The clocks of the system are initially synchronized to within δ_o.
4. The system executes the algorithm every R seconds and provides enough CPU time for the algorithm to complete.

Each of these assumptions must be established to a confidence level consistent with the reliability requirements. Thus, although life testing of the system as a whole can be avoided, life testing must effectively be performed on certain system parameters. However, the behavior of these low-level components of the system are typically far less complex than the system as a whole, and the components are thus easier to validate. Furthermore, the formal proof provides a precise statement of exactly which properties of the system must be measured.

Measurement of System Parameters

The mathematical theorem defines the worst-case clock skew in terms of certain system parameters. These parameters are specific to each implementation of the algorithm. The SIFT system was originally implemented on seven Bendix BDX-930 processors. However, extracting synchronization data from that system is currently difficult. To investigate methods of validation, the SIFT synchronization algorithm was implemented on four VAX-11/750 computers in AIRLAB. Some of the system parameters are directly measurable on the VAX system. Others, such as the maximum read error ϵ must be indirectly measured. The results of these measurements are analyzed in a subsequent section, after the explanation of the theoretical method.

The first parameter to be discussed is ϵ, because it is the most difficult to measure and plays a significant role in the system performance. As defined previously, ϵ is an upper bound on $|e_{qp}|$ over all processor pairs and over all time, where $e_{qp} = r_p^{(i)}(T_0 + \Delta_{qp}) - r_q^{(i)}(T_0)$. Unfortunately, e_{qp} is defined in terms of real time rather than observable clock time. In the appendix, the following formula defining e_{qp} in terms of observable clock times is shown to be a highly accurate approximation of the theoretical e_{qp}:

$$A_{qp} + e_{qp} = \Delta_{qp}$$

and

$$|e_{qp}| < \epsilon$$

where A_{qp} is the difference between clocks p and q at real time t (i.e., actual skew at t):

$$A_{qp}(t) = C_p(t) - C_q(t)$$

It is still necessary to characterize e_{qp} in a system context. In the system, a processor p reads a processor q's clock by the following method: At a prespecified time, processor q reads its clock and transmits the value $C_q(t_1)$ to processor p. Upon receiving the message, processor p immediately reads its clock to obtain $C_p(t_2)$. As shown in figure 4, if the exact communication delay X_{qp} were known, then the exact skew A_{qp} could be calculated by

$$A_{qp} = C_p(t_2) - C_q(t_1) - X_{qp}$$

Thus, the designer of the synchronization system chooses a value v approximately equal to $E(X_{qp})$ to be used by the system to compute an apparent skew Δ_{qp} by the following formula:

$$\Delta_{qp} = C_p(t_2) - C_q(t_1) - v$$

Because the communication delay is variable, each calculation of Δ_{qp} is subject to an error of $X_{qp} - v$. There are two components to this error, and they are shown in the following equation:

$$e_{qp} = X_{qp} - v = [X_{qp} - E(X_{qp})] + \mu$$

where

$$\mu = E(X_{qp}) - v$$

The first component $[X_{qp} - E(X_{qp})]$ is the variation due to the random nature of the communication. The second component μ is a constant offset due to the system designer's error in choosing v. Also, it follows from the above formula that $E(e_{qp}) = \mu$.

The distribution of e_{qp} may be obtained from measurements of the one-way communication delays. However, this requires some form of special hardware. In the AIRLAB VAX system, a special Pulse Network was used to measure this delay. The delay for sending a pulse is considerably less variable than the communication delay for sending a message. Thus, reasonably accurate measurements of the communication delay were made by the following method: One processor's clock was read, and the value was sent to a second processor. When the second processor received the message, a pulse was immediately sent over the Pulse Network to the first processor. When the first processor received the pulse, its clock was read again. By subtracting the first clock value from the second clock value, the communication delay plus the pulse delay were measured. Subtracting an estimate of the mean pulse delay from this value provides an accurate measurement of the communication delay. Figure 5 is a histogram of 2000 such estimates of X_{qp}.

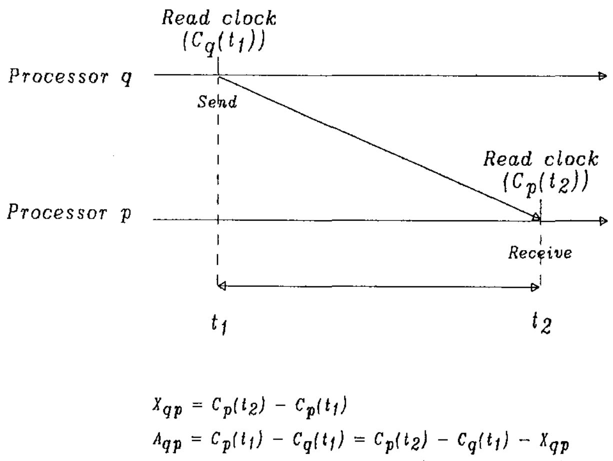

Figure 4. Calculation of actual skew A_{qp}.

Next, a method is discussed that provides a means of estimating both ϵ and ρ using the internal state information of the synchronization system. Thus, the method requires no special external measuring hardware. The physics of crystal clocks dictates that the drift rate ρ_{qp} between any two clocks q and p is constant over time. Thus, if the system is run without synchronization, the model

$$\Delta_{qp}(T) = \delta_{qp}(0) + \rho_{qp}T + e_{qp}(T)$$

describes the system, where $\delta_{qp}(0)$ is the initial skew between clocks q and p at time 0, and ρ_{qp} is the drift rate between clocks q and p. The values of $\Delta_{qp}(T)$ are directly observable from the various processors memories. Since the Δ_{qp}'s are computed every R seconds, the data consist of $\Delta_{qp}(T^{(i)})$, where $T^{(i)} = T(0) + iR$ and $i = 1, n$. Thus, a linear least-squares analysis can be used to estimate the parameters $\delta_{qp}(0)$ and ρ_{qp}. (See fig. 6.) The residuals from the regression can be represented by ϕ, and the equation can be written as:

$$\Delta_{qp}(T) = \hat{\alpha} + \hat{\beta}T + \phi$$

If the e_{qp}'s are distributed with mean zero, then $\hat{\beta}$ is an estimate of ρ_{qp}, and the residuals have approximately the same distribution as the e_{qp}'s. However, this may not be the case. Suppose that $E(e_{qp}) = \mu \neq 0$. Then, $\alpha = \delta_{qp}(0) + \mu$, and the residuals have approximately the same distribution as $e_{qp} - \mu$. Thus, a histogram of e_{qp}'s can be obtained by adding μ to the residuals.

Next, a simple independent method to estimate μ is

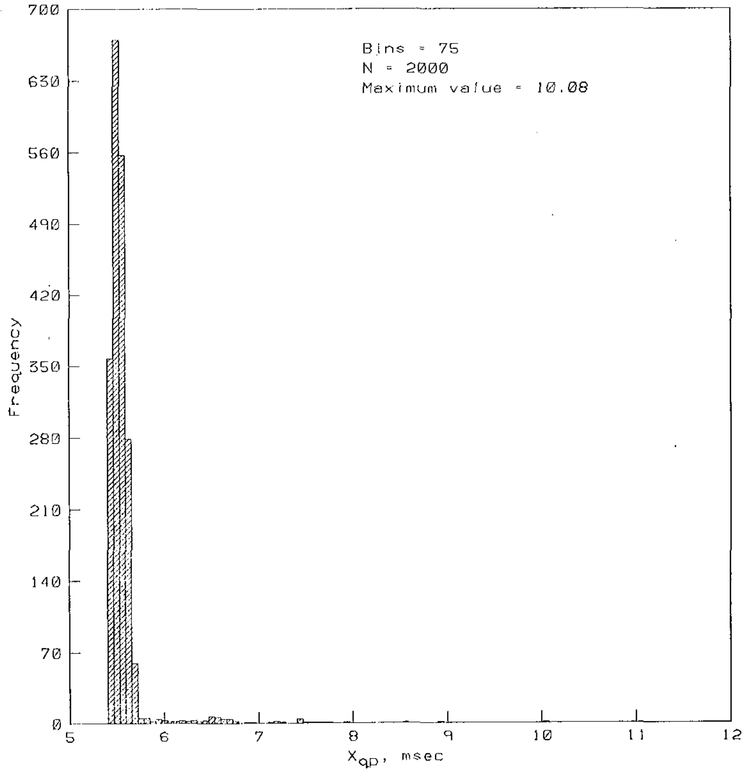

Figure 5. Histogram of communication delays.

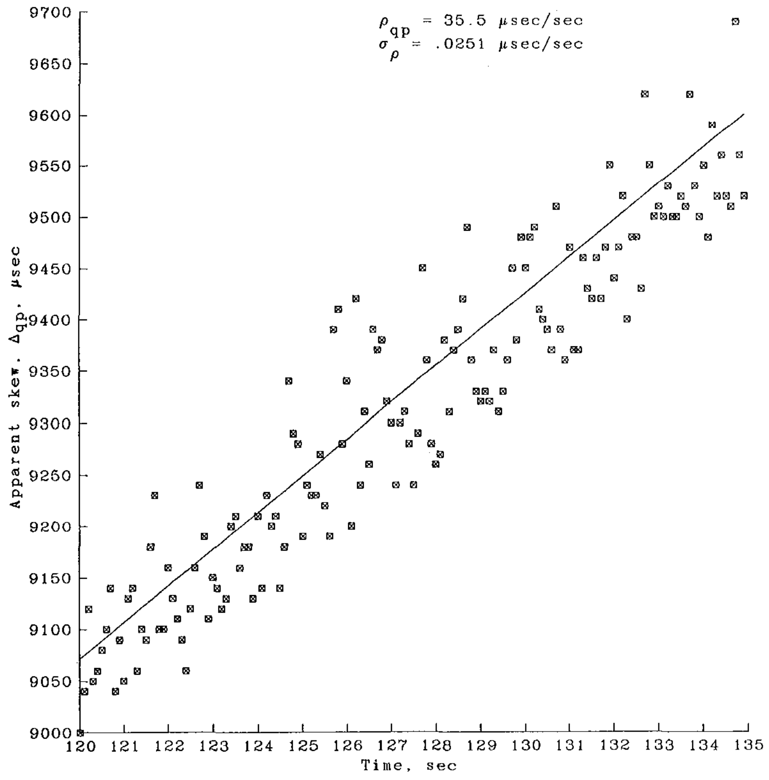

Figure 6. Least-squares fit to measured skews Δ_{qp}.

presented. Since $\Delta_{qp} = A_{qp} + e_{qp}$,

$$\Delta_{qp} + \Delta_{pq} = e_{qp} + e_{pq} + A_{qp} + A_{pq}$$

Furthermore, since $A_{qp} = -A_{pq}$,

$$\Delta_{qp} + \Delta_{pq} = e_{qp} + e_{pq}$$

Thus,

$$E(e_{qp}) = E\left(\frac{\Delta_{qp} + \Delta_{pq}}{2}\right)$$

since $E(e_{qp}) = E(e_{pq})$. Therefore, μ can be estimated by the sample average of several observations of $(\Delta_{qp} + \Delta_{pq})/2$.

In a properly "tuned" synchronization system, $\mu = 0$. When $\mu = 0$ in a system, the absolute value of the read error is minimized. The following method may be used to tune the synchronization system. As shown earlier, the e_{qp}'s consist of two components—the deviation from the mean communication time, and the constant offset μ. Since μ can be expressed as follows:

$$\mu = E(X_{qp}) - v = E(X_{qp} - v) = E(e_{qp})$$

the constant offset can be eliminated by simply adding the above estimate of μ to v. The code in the implementation may thus be corrected to use this new value of v.

Next, an upper bound on $|e_{qp}|$ must be estimated from e_{qp} histograms obtained from either method. A statistical approach to this problem is discussed in the following section.

A Statistical Approach to Estimating ϵ

In this section, the problem of estimating an upper bound of $|e_{qp}|$ from histograms of experimental data is addressed. The motivation for this exercise is to statistically estimate the parameter ϵ with confidence consistent with the reliability requirement of a probability of failure not to exceed 10^{-9} for a 10-hour flight. The quantile of the e_{qp} distribution to which ϵ must be equated is specific to each system and may be determined by a reliability analysis of the system. This quantile is usually on the order of $1 - 10^{-9}$. Unfortunately, the traditional method of estimating a quantile requires an exorbitant sample size in this case, as shown in the following paragraph.

Let $x_1, x_2, x_3, \ldots, x_n$ be a random sample from a distribution $F(x)$. The pth sample quantile is the number $\hat{\xi}_p$, such that the fraction of x_i's that are less than $\hat{\xi}_p$ is $\leq p$, and the fraction of x_i's greater than $\hat{\xi}_p$ is $\leq 1-p$. Thus, to estimate the theoretical ξ_p quantile by this technique, one must observe at least one x_i greater than the ξ_p quantile. This requires an extremely large sample size as may be seen by the following calculation. Let $Z = \max(x_1, x_2, x_3, \ldots, x_n)$. Then,

$$\begin{aligned}\text{Prob}(Z > \xi_p) &= 1 - \text{Prob}(Z < \xi_p)\\ &= 1 - \text{Prob}(x_1 < \xi_p,\\ &\qquad x_2 < \xi_p, \ldots, x_n < \xi_p)\\ &= 1 - \prod_i^n \text{Prob}(x_i < \xi_p)\\ &= 1 - p^n\end{aligned}$$

Thus, to be $(1 - \alpha) \times 100$ percent confident that an observation will exceed ξ_p, n must be chosen such that

$$1 - p^n = 1 - \alpha$$

or

$$n = \ln(\alpha)/\ln(p)$$

For $\alpha = 0.75$ and $p = 1 - 10^{-9}$, $n = 2.876 \times 10^8$ observations. Such a large sample size is usually impractical. The only remaining choice is to assume an underlying parametric model of the distribution X or to assume some special properties of the distribution. Sometimes a parametric model can be theoretically derived from an analysis of the communication system. Often, however, such a model is not obtainable. Furthermore, any statistical inference made would be strongly dependent on the assumption that the experimental data were generated from the chosen parametric family of distributions. The danger inherent in such a method can be seen in figure 7. In this figure, an attempt to fit a random sample of X_{qp} to a 3-parameter Weibull distribution is illustrated. (See ref. 4.) Although this is a very large family of distributions, an unfortunate situation is observed. The estimated $1 - 10^{-9}$ quantile is smaller than several observed data values! (See fig. 7.) Even when the fit is statistically very good, the model may prove inappropriate for inference with respect to the tail of the distribution. Clearly, some other method of estimating properties of the tail of a distribution is needed.

Although the estimation of the maximum was intractable when no assumptions were made about the underlying distribution, certain minimal assumptions can simplify the estimation problem considerably. The statistical theory presented next was developed by Weissman. (See ref. 2.) This theory enables the estimation of large quantiles of the underlying population distribution from the k largest observations of a random sample. The major assumption of the method is that the underlying distribution function F is in the "domain of attraction" of some known distribution function G. This property is satisfied by a large class of distribution functions (e.g., the gamma, exponential, Weibull, normal, lognormal, and logistic). Therefore, the method is essentially nonparametric. Mathemati-

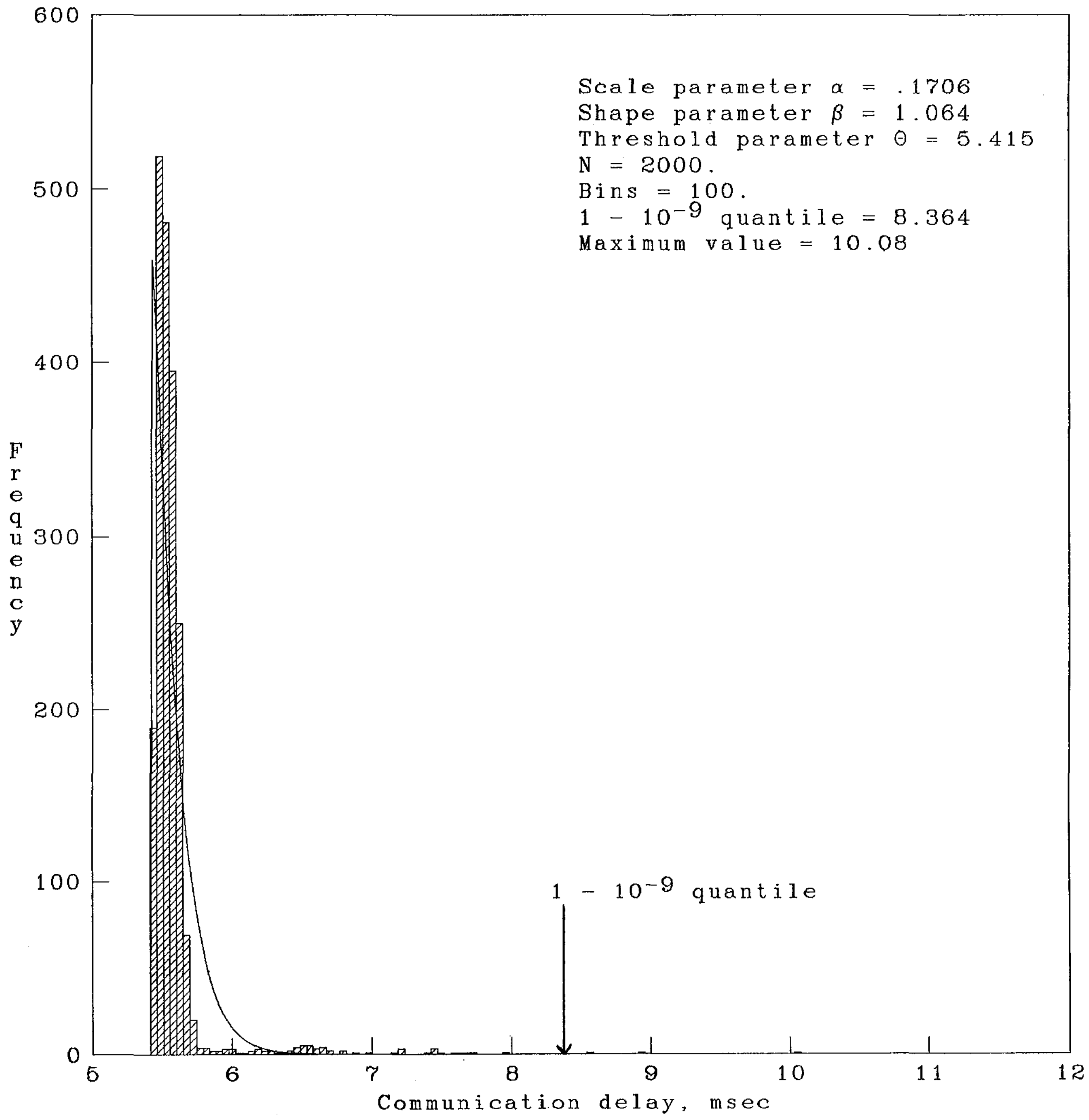

Figure 7.- Weibull fit to communication delay histogram.

cally this assumption is as follows: If $x_1, x_2, x_3, \ldots, x_n$ is a random sample from a distribution $F(x)$ and Z_n is the largest observed x, then the distribution function for Z_n is $F^n(x)$. If there exist sequences $a_n > 0$ and b_n for all n and a distribution function G such that

$$F^n(a_n x + b_n) \rightarrow G(x) \quad \text{as} \quad n \rightarrow \infty$$

for all values of x where G is continuous, then $G(x)$ is an extremal distribution function and $F(x)$ lies in its "domain of attraction." (See ref. 3.) This distribution function G must be from one of the following families of distributions:

1. $\text{LAMBDA}(x) = \exp(-e^{-x}) \qquad (-\infty < x < \infty)$
2. $\text{PHI}_a(x) = \exp(-x^{-a}) \qquad (x > 0, a > 0)$
3. $\text{PSI}_a(x) = \exp(-(-x)^a) \qquad (x < 0, a > 0)$

The theory developed by Weissman makes possible the estimation of large quantiles of the underlying distribution. His method is based on the result that the k largest order statistics, when normalized by constants a_n and b_n, converge in distribution to a k-dimensional extremal variate (a vector of variables distributed as the ordered times at which events occur in a nonhomogeneous Poisson process). The normalizing constants are treated as unknown parameters indexing the limiting distribution and are estimated by the method of maximum likelihood. That is, the estimation problem is solved by basing the estimates on the limiting distribution rather than on the underlying distribution of the e_{qp}. As indicated subsequently, the quantiles of the underlying distribution may be represented in terms of the parameters a_n and b_n, thereby obtaining maximum likelihood estimates of the quantiles of the underlying distribution.

For example, by choosing ϵ to be the $1 - 10^{-9}$ quantile, the probability of the system exceeding the design assumption is 10^{-9}. The method provides a simple method of calculating the large quantiles, once the limiting distribution family has been determined (i.e., one of the three listed previously in this section). Fortunately, simple statistical tests are available for making this determination. These are discussed subsequently. The Weissman technique only uses the largest k values of the random sample. The choice of the k is arbitrary, although it should be small in comparison with the sample size (e.g., $k = 10$ and $n = 1000$). The value of k is chosen prior to the examination of the data.

Once k is chosen and the limiting distribution is determined, one of the following calculations is performed depending on the limiting distribution. In each case below $X_{1n} \geq X_{2n} \geq \ldots \geq X_{kn} \geq \ldots \geq X_{nn}$ represents the order statistics of the random sample.

CASE 1 (G = LAMBDA):

$$1 - c/n \text{ quantile} = a_n[-\ln(c)] + b_n$$

where

$$a_n = \left[\sum_{i=1}^{k}(X_{in})/k\right] - X_{kn}$$

and

$$b_n = a_n \ln(k) + X_{kn}$$

CASE 2 (G = PHI):

$$1 - c/n \text{ quantile} = (k/c)^{1/a} X_{kn}$$

where

$$1/a = \left[\sum_{i=1}^{k}\ln(X_{in})/k\right] - \ln(X_{kn})$$

CASE 3 (G = PSI):

Since case 3 applies only to negative x, it is not appropriate for this application and is not discussed in detail here.

The remaining problem is the determination of the limiting distribution G. It is possible to test the hypothesis that G = LAMBDA by testing whether the set of normalized spacings $D_{1n}, 2D_{2n}, \ldots, (k-1)D_{(k-1)n}$ are independent, identically distributed exponential random variables, with $D_{in} = X_{in} - X_{(i+1)n}$. Similarly, the hypothesis G = PHI can be tested by determining whether the normalized spacings of $\ln(X_{in})$ are independent, identically distributed exponential random variables. The Gini statistic can be used for these tests. (See ref. 4.) The Gini statistic G_s is calculated as follows:

$$G_s = \sum_{i=1}^{s}\sum_{j=1}^{s} |y_i - y_j| / 2s(s-1)\bar{y}$$

where $y_1, y_2, \ldots, y_s$ is the random sample being tested for exponentiality. The statistic G_s necessarily lies between 0 and 1, with values near 0 or 1 indicating nonexponentiality. For values of s larger than 20, the standardized form of the Gini statistic,

$$W_s = [12(s-1)]^{1/2}(G_s - 0.5)$$
$$\approx \text{the Standard Normal, N}(0,1)$$

may be used to determine the significance level. Thus, for example, a W_s value of 2.96 indicates an observed significance level of 1 percent, and permits the rejection of the hypothesis that the distribution is exponential with 99-percent confidence.

A Statistical Approach to Estimating ρ

In this section, a method is presented for estimating ρ, the maximum drift rate between any two clocks in the system, such that if ρ_{qp} represents the drift rate between clocks q and p, then

$$\rho_{qp} < \rho$$

for all clocks q and p. This is precisely design assumption 1 discussed previously. To determine the probability that this design assumption is violated, it is necessary to calculate the probability that one or more ρ_{qp} exceed the estimated upper bound $\hat{\rho}$, or

$$\text{Prob}(\rho_{qp} > \hat{\rho}) \text{ for some } q \text{ and } p$$

If there are n processors in the system being validated, then there are $n_c = \binom{n}{2}$ drift rates between processor pairs. For simplicity, these are referred to as ρ_i, where $i = 1$ to n_c.

Using the linear regression analysis on the $\Delta_{qp}(T^{(i)})$ data described previously, a set of estimates

$$\{(\hat{\rho}_i, \hat{\sigma}_i^2) \mid i = 1, n_c\}$$

can be obtained, and $\hat{\rho}_i$ is an estimate of the drift rate between processor pair i and $\hat{\sigma}_i^2$ is an estimate of the variance of $\hat{\rho}_i$.

From the experimental data, an estimate of the upper bound of the drift rates $\hat{\rho}$ must be determined such that $\text{Prob}(\max(\rho_i) > \hat{\rho}) = \alpha$ is sufficiently small.

The maximum drift rate ρ may be estimated as follows:

$$\hat{\rho} = \max(u_i)$$

where u_i is defined by:

$$\text{Prob}(\rho_i > u_i) = \sqrt[n_c]{1 - \alpha}$$

The following theorem shows that this estimate is adequate:

Theorem: $\text{Prob}(\max(\rho_i) < \hat{\rho}) \geq 1 - \alpha$.

Proof:

$$\begin{aligned}
&\text{Prob}(\max(\rho_i) < \hat{\rho}) \\
&\quad = \text{Prob}(\max(\rho_i) < \max(u_i)) \\
&\quad = \text{Prob}(\rho_1 < \max(u_i), \rho_2 < \max(u_i), \ldots, \\
&\qquad \rho_n < \max(u_i)) \\
&\quad \geq \text{Prob}(\rho_1 < u_1, \rho_2 < u_2, \ldots, \rho_n < u_n) \\
&\quad = \prod_i^{n_c} \sqrt[n_c]{1 - \alpha} \\
&\quad = 1 - \alpha
\end{aligned}$$

The values of u_i are easily obtained by using the following formula:

$$u_i = \hat{\rho}_i + t(\nu, \theta)\hat{\sigma}_i$$

where

ν	$= n_s - 2$
n_s	= number of data points used in regression analysis to obtain $\hat{\rho}_i$ and $\hat{\sigma}_i$
θ	$= \sqrt[n_c]{(1 - \alpha)}$
$t(\nu, \theta)$	= θ percentage point of student's t distribution with ν degrees of freedom

For $n_s > 100$, $t(\nu, \theta)$ may be replaced by a percentage point of the standard normal distribution. The following approximation formula for the normal distribution $F(z)$ (see ref. 5) is useful for small values of α (i.e., large z):

$$1 - \alpha = F(z) = 1 - \frac{1}{z\sqrt{2\pi}} \exp(-0.5z^2)$$

Validation of the AIRLAB Experimental System

In this section, the methods developed in the previous sections are combined into a complete validation method. This validation method is illustrated by application to measurements made on the AIRLAB experimental synchronization system. As described previously, this system consists of four communicating VAX-11/750 computers. These processors exchange clock values and synchronize themselves using the SIFT fault-tolerant clock synchronization algorithm. As discussed previously, the four major design assumptions which must be validated are as follows:

1. The maximum drift rate between any two working clocks is $< \rho$.
2. If two processors are nonfaulty then one processor can read another processor's clock to within an error of ϵ.
3. The clocks of the system are initially synchronized to within δ_o.
4. The system executes the algorithm every R seconds and provides enough CPU time for the algorithm to complete.

Design assumption 3 corresponds to a process that would occur at system initialization. Since this process occurs before system operation, while the aircraft is on the ground, it need not be fault tolerant. If the initialization process fails, it can be restarted. Detection of such a failure is not difficult. Design assumption 4 is intimately connected with the performance characteristics of the test-specimen operating-system scheduler. Analysis of the operating-system scheduler is strongly dependent on the scheduling method employed in the system. Therefore, it is not possible to present a generic method for validation of this assumption. Hence, only the first two design assumptions are analyzed in detail.

The validation method depends fundamentally on a mathematical reliability analysis of the system. Such a reliability analysis must include the following three probabilities of failure:

p_h	= Prob(one or more hardware component failures in a processor)
p_1	= Prob(design assumption 1 being violated)
p_2	= Prob(design assumption 2 being violated)

The probability p_h may be obtained from a military standard 217D analysis of component failure data. This analysis method is well-known and therefore is not discussed here. A processor failure rate of 10^{-5}/hour is assumed. The probability p_1 is the measurement error in determining the upper bound ρ. This probability may be made arbitrarily small by increasing the bound and/or increasing the accuracy of the measurements (e.g., reducing $\hat{\sigma}_\rho$ by using a larger sample size). The probability of failure p_2 arises from the stochastic nature of the communication system that is used to read another processor's clock. This probability may also be reduced by increasing the bound ϵ; however, this is done at the expense of increasing the estimated maximum clock skew $\hat{\delta}$. This trade-off is discussed in detail in the section "Additional Observations About the Clock Synchronization Algorithm."

The validation method entails the following steps:

1. Determine the upper bound $\hat{\rho}$ such that Prob($\hat{\rho} < \rho$) is negligible in comparison with p_h.
2. Determine the probability quantile needed for ϵ from a reliability analysis of the system.
3. Estimate this quantile from experimental data and use this value as the maximum read error ϵ.
4. Compute the maximum clock skew from $\hat{\rho}$ and $\hat{\epsilon}$.
5. Determine whether this maximum clock skew value exceeds the value assumed in the system design.

Thus, $\hat{\rho}$ and $\hat{\epsilon}$ are chosen large enough to meet the system reliability requirements. Using these bounds, a theoretical maximum clock skew is calculated. If the maximum clock skew assumed in the design of the system is not greater than the theoretical maximum value, then the system is validated.

Validation Step 1

In this validation step, $\hat{\rho}$ must be determined such that p_1 is small in comparison with p_h. This approach is desirable because p_1 is the probability of a measurement error rather than an intrinsic failure mode of the system. As mentioned previously, p_1 can be made arbitrarily small by improving the measurement technique (i.e., reducing $\hat{\sigma}_\rho$).

A regression analysis of the $\Delta_{qp}(T^{(i)})$ experimental data produced the following table:

Processor pair	i	$\hat{\rho}_i$	$\hat{\sigma}_i$	$\hat{u}_i$
1,2	1	30.02	0.2687	31.462
1,3	2	9.01	.0245	9.143
1,4	3	35.50	.0251	35.635
2,3	4	14.92	.0954	15.432
2,4	5	5.48	.0390	5.686
3,4	6	40.97	.0851	41.427

The values of $\hat{u}_i$ were calculated using $\hat{u}_i = \hat{\rho}_i + t(\nu,\theta)\hat{\sigma}_i$, where $\theta = \sqrt[n_c]{1-10^{-7}} = 1 - 1.667 \times 10^{-8}$ and $\nu = n_s - 2 = 1998$. Thus, $p_1 = 10^{-7}$, which is small relative to p_h. The maximum drift rate ρ is estimated by $\max(u_i)$. Thus,

$$\hat{\rho} = 41.427$$

Validation Step 2

In this step the required quantile for ϵ must be determined from a reliability analysis of the system. Typically, a detailed Markov model is necessary to calculate the reliability of a fault-tolerant system. However, for simplicity, it is assumed here that there is no sparing capability in the system, and thus a simple combinatorial analysis can be used to compute the reliability of the system. Since the algorithm can tolerate m processor failures, the probability of $m+1$ processor failures during the flight must be determined. The system is thus a 2-out-of-4 system.

The probability of a processor failure is as follows:

$$p = p_h + p_1 + p_2$$

Therefore, the probability of a system failure during a mission of length T is

$$\begin{aligned} p_{\text{sys}} &= 1 - \text{Prob(no failures)} - \text{Prob(1 failure)} \\ &= 1 - \binom{4}{0}p^0(1-p)^4 - \binom{4}{1}p^1(1-p)^3 \\ &= 6p^2 - 8p^3 + 3p^4 \\ &\approx 6p^2 \end{aligned}$$

Given a reliability requirement of $p_{\text{sys}} = 10^{-9}$, a processor failure rate of $p_h = 10^{-5}$/hour and $p_1 = 10^{-7}$, we have

$$\begin{aligned} p_2 &= \text{Prob(one or more read errors} > \epsilon \\ &\qquad \text{occurring on a specific processor during} \\ &\qquad \text{a mission of length } T) \\ &= \sqrt{p_{\text{sys}}/6} - p_h - p_1 = 2.809 \times 10^{-6} \end{aligned}$$

Since the bound ϵ refers to a single clock read, the number of clock reads during a mission of length T must be determined in order to calculate the quantile needed for ϵ.

Defining p_ϵ as

$$p_\epsilon = \text{Prob(obtaining a read error} > \epsilon \text{ during a single clock read)}$$

the probability p_2 is then easily expressed in terms of p_ϵ as follows:

$$p_2 = 1 - \binom{n}{0} p_\epsilon^0 (1 - p_\epsilon)^n$$

where

$n = (N - 1)T/R$ (i.e., the number of clock reads a specific processor makes during a mission of length T)

and

T	mission time
R	synchronization period
N	number of processors in system

Using the Poisson approximation to the binomial,

$$p_2 = 1 - \exp(-np_\epsilon)$$

Furthermore, by a Taylor series approximation (valid because $np_\epsilon \ll 1$),

$$p_2 = np_\epsilon = (N - 1)(T/R)p_\epsilon$$

Using $N = 4$, $R = 30$ sec, and $T = 10$ hours, the probability p_ϵ is determined as follows:

$$p_\epsilon = 7.805 \times 10^{-10}$$

This analysis assumes independence of clock read-error failures. The design proof has thus reduced the strong assumption of independent clock failure to independent communication. This analysis makes a strong case for avoiding contention-based communication protocols in a fault-tolerant architecture.

Validation Step 3

The third step in validating the system under investigation is to estimate the $1 - p_\epsilon$ quantile of the read-error distribution. Two methods were developed in the preceding sections to obtain a histogram of the clock read errors e_{qp}. Figure 8 is a histogram of $|e_{qp}| = |X_{qp} - v|$ obtained from direct measurements of the one-way communication times. These data are used to illustrate the determination of $\hat{\epsilon}$.

As discussed in a preceding section, the upper bound ϵ is determined using Weissman's technique. Prior to examination of the data, k was chosen to be 20 (i.e., 1 percent of the sample), as recommended by Weissman. The experimental data best support LAMBDA as the limiting distribution; however, there was no clear rejection of either of the limiting distributions. The standardized form of the Gini statistic W_{k-1} applied to the $k - 1$ normalized spacings of the k largest observations from the sample, and the corresponding observed significance levels for the tests were as follows:

Limiting distribution	W_{19}	Observed significance level, percent
LAMBDA	0.5262	59.2
PHI	1.449	14.7

The inability to choose the limiting distribution with precision is of some concern here. Examining the test results for various values of k provides additional insight into discerning the limiting-distribution family. In figure 9, the standardized Gini statistics for the LAMBDA and PHI tests using various values of k are plotted. The tests show a consistent tendency toward selection of the LAMBDA distribution. In fact, for some values of k there is strong rejection of the PHI and strong acceptance of the LAMBDA. As k becomes larger, both models are eventually rejected, because Weissman's theory applies only to the tail of a distribution. Although the additional information obtained by varying k intuitively leads to a choice of the LAMBDA distribution, how to use such information has not been formalized statistically.

An alternate solution to the problem of discerning the limiting-distribution family is to calculate the $1-p_\epsilon$ quantile from both family models and to use the most conservative value. However, sometimes the poorly fitting model gives astronomical values—leading to unacceptable answers. An alternative approach is to pursue additional statistical methods to determine the limiting-distribution family. Such methods are not presented here.

The remaining calculations are performed with the assumption that LAMBDA is the correct limiting distribution. Using the LAMBDA case analysis, the $1-p_\epsilon$ quantile was estimated to be 15.383 msec. The combinatorial analysis has shown that $\hat{\epsilon}$ must be at least the $1 - p_\epsilon$ quantile to meet the system reliability requirements. Using this quantile to estimate the upper bound ϵ,

$$\hat{\epsilon} = 15.383 \text{ msec}$$

The conservative nature of this estimate can be seen by comparison with the maximum observation, 4.49 msec.

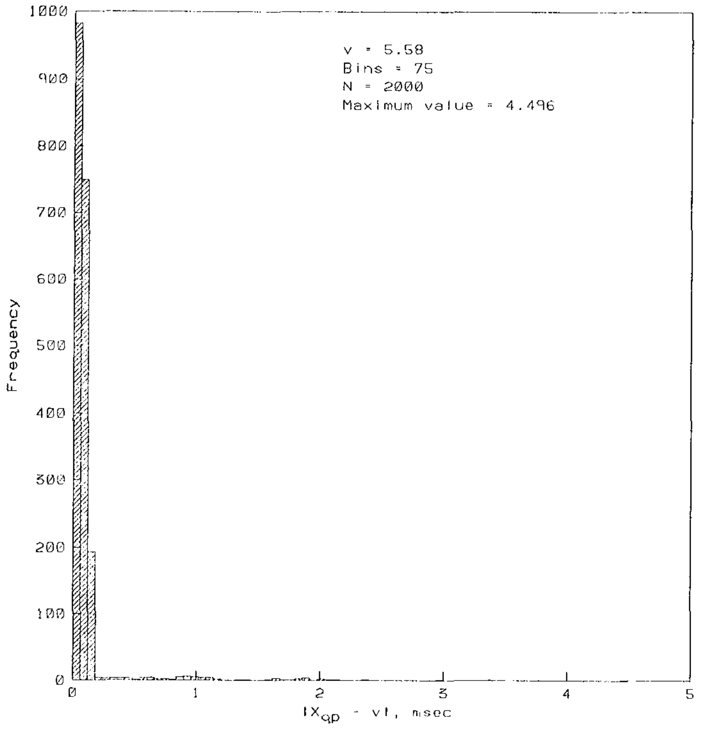

Figure 8. Histogram of $|e_{qp}| = |X_{qp} - v|$.

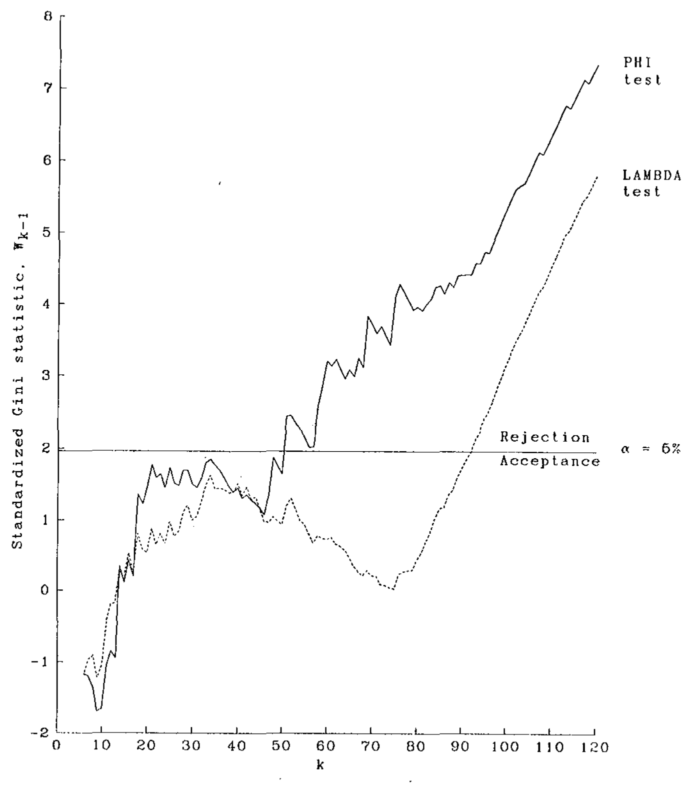

Figure 9. Results of tests for limiting-distribution family.

Validation Step 4

The estimated values of ρ and ϵ must be inserted into the theoretical expression for maximum clock skew from the design-proof theorem:

$$\delta \geq \frac{N}{N-3m}\left\{2\epsilon + \rho\left[R + 2\left(\frac{N-m}{N}\right)S\right]\right\},$$
$$\delta \geq \delta_o + \rho R,$$
$$\delta \ll R,$$

and

$$\delta \ll \epsilon/\rho$$

The directly measured and indirectly estimated values of the system parameters are as follows:

$$N = 4$$
$$m = 1$$
$$R = 30 \text{ sec}$$
$$S = 615.334 \text{ msec}$$
$$\epsilon = 15.383 \text{ msec}$$
$$\rho = 41.42657\ \mu\text{sec/sec}$$

Using these values, the maximum clock skew δ can be computed as follows:

$$\begin{aligned}\delta &= [N/(N-3m)]\{2\epsilon + \rho[R + 2(N-m)S/N]\}\\ &= 123.061 + 1.65706 \times 10^{-4}(30\,000 + 3(615.334)/2)\\ &= 123.061 + 5.124\\ &= 128.185 \text{ msec}\end{aligned}$$

Thus, the clocks remain synchronized to within 128.185 msec with probability not less than $1 - 10^{-9}$ if the synchronization period is 30 seconds. The contribution of the second term is small relative to the first. This reveals that, in this implementation, the clocks are much more accurate than the interprocess communication subsystem.

Validation Step 5

As discussed previously, the communication subsystem of a real-time system depends critically on synchronization being maintained within a certain bound. If the calculated skew is less than the bound used in the design of the communication subsystem, then the synchronization system has been validated. Otherwise, the real-time system must be redesigned if the reliability requirements are to be met. This may be accomplished by either slowing down the communications system (i.e., waiting longer for interprocess data) or making improvements to reduce ρ and/or ϵ. The trade-off between performance and reliability is explored in detail in the next section.

These validation steps can be reversed to compute system reliability, given a specific design value for the maximum clock skew δ. Essentially, ρ is estimated as described previously such that $p_1 = 10^{-7}$. Then ϵ is computed from the formula of the theorem using these values of ρ and δ. Next, the probability that ϵ is exceeded, p_2 can be computed, and hence, p_{sys} can be determined. Thus, the system reliability is determined, given a specific design choice for the maximum clock skew.

Additional Observations About the Clock Synchronization Algorithm

The synchronization algorithm is executed periodically and utilizes CPU time during each execution. The major component of the execution time is the time required to read the clock of every other processor in the system. In SIFT, each processor's clock value is broadcast during a window of time allocated to it. There are N such windows, one for each processor in the system. All other processors wait during this window to receive the broadcast clock value. To accommodate the worst-case situation, each window must be at least $B+\delta$ seconds long, where B is the maximum broadcast time (i.e., $v + \epsilon$) and δ is the maximum clock skew. Hence, the execution time for the clock synchronization algorithm can be represented as

$$S = N(\delta + B) + K$$

where K is the time needed to compute the correction factor and to correct the clock. The execution time of the synchronization task S contributes to the inability to synchronize perfectly, because the clocks continue to drift apart while the synchronization task is executing. Thus, the equations defining δ and S are coupled as follows:

$$S = S(N, \delta, B, K)$$
$$\delta = (N, m, \epsilon, \rho, R, S)$$

Also, ϵ is dependent on the system reliability requirement p_{sys} and the synchronization period R. Therefore,

$$\epsilon = \epsilon(p_{\text{sys}}, R)$$

Although an algebraic solution is tedious, δ can easily be numerically determined as a function of the synchronization period R or the fraction of time spent synchronizing S/R. It is more informative, however, to relate the experimental results to the performance of a hypothetical real-time communication system. A real-time communication system relies on the synchronization task, as shown in figure 1. The minimum communication time is $B + \delta$, since the system must wait at least this long to insure that the data value has arrived before accessing it. Since $B = v + \epsilon$, the minimum communication time can be expressed as $(v + \epsilon + \delta)$. This minimum communication time represents the impact of the synchronization system on the performance of the real-time system. Thus, the performance of the real-time communication system is a function of the system reliability. This performance-reliability trade-off is illustrated in figure 10. As expected, as the reliability requirement is relaxed, the performance of the system increases. Also, the performance is a function of the synchronization period R or the fraction of time spent synchronizing S/R. This performance-overhead trade-off is illustrated in figures 11 and 12.

Concluding Remarks

The validation method presented in this paper is an exploitation of the precision with which the formal proof method reduces the complexity of the system to verifiable axioms about the system behavior. Although the proof process itself is very costly, it is extremely valuable when attempting to validate the crucial synchronization subsystem. The validation method introduced in this paper is essentially to: (1) perform a design proof of the synchronization algorithm under the assumption of low-level system behavior axioms, (2) perform a code proof of the synchronization code, and (3) experimentally estimate the probability that the system behavior axioms will be violated and include these failure probabilities into a reliability analysis of the system. The Software Implemented Fault Tolerance (SIFT) synchronization design proof provided the basis for step (1). Step (2) has not yet been attempted, but will be performed under NASA contract NAS1-17067. Step (3) is explored in detail in this paper. Although the validation theory is not directly applied to the data obtained from the SIFT hardware, the validation theory is applied to data obtained from an experimental system in the Langley Avionics Integration Research Laboratory (AIRLAB). The purpose of this paper is to define a validation method rather than specifically validate the SIFT synchronization subsystem. After the development of the SIFT data-retrieval system in early 1984, this theory will be applied to the SIFT hardware.

The design proof process reduces the performance of the clock synchronization algorithm to an algebraic expression of certain system parameters. These parameters, which are defined by formal axioms, represent worst-case bounds on system performance. By simple combinatorial analysis, the system reliability requirements can be translated to reliability requirements on these bounds. Because the estimation of a bound of a random variable is required, statistical methods applicable to the tail of a distribution are employed. The estimated parameters are substituted into the algebraic expression which calculates the worst-case performance of the synchronization system. This estimated worst-case performance (given the specified reliability constraints) is compared against the system design value. This validation process thus yields estimates of both performance and reliability.

Langley Research Center
National Aeronautics and Space Administration
Hampton, VA 23665
July 18, 1984

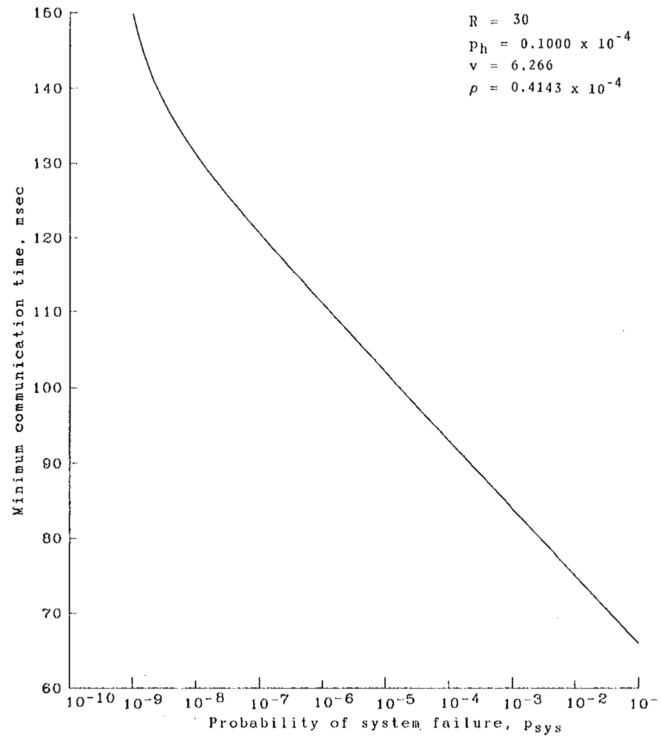

Figure 10. Performance-reliability trade-off.

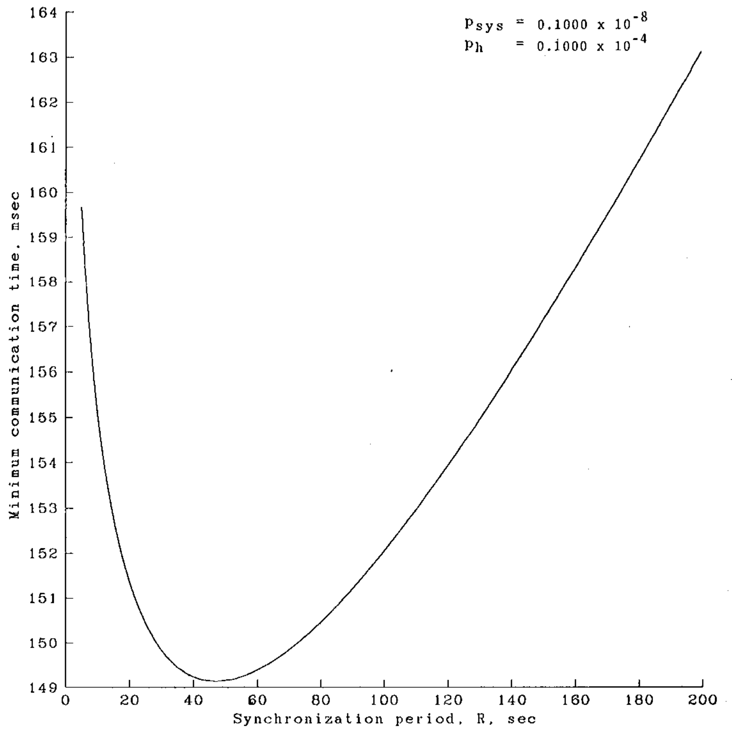

Figure 11. Performance as a function of synchronization period.

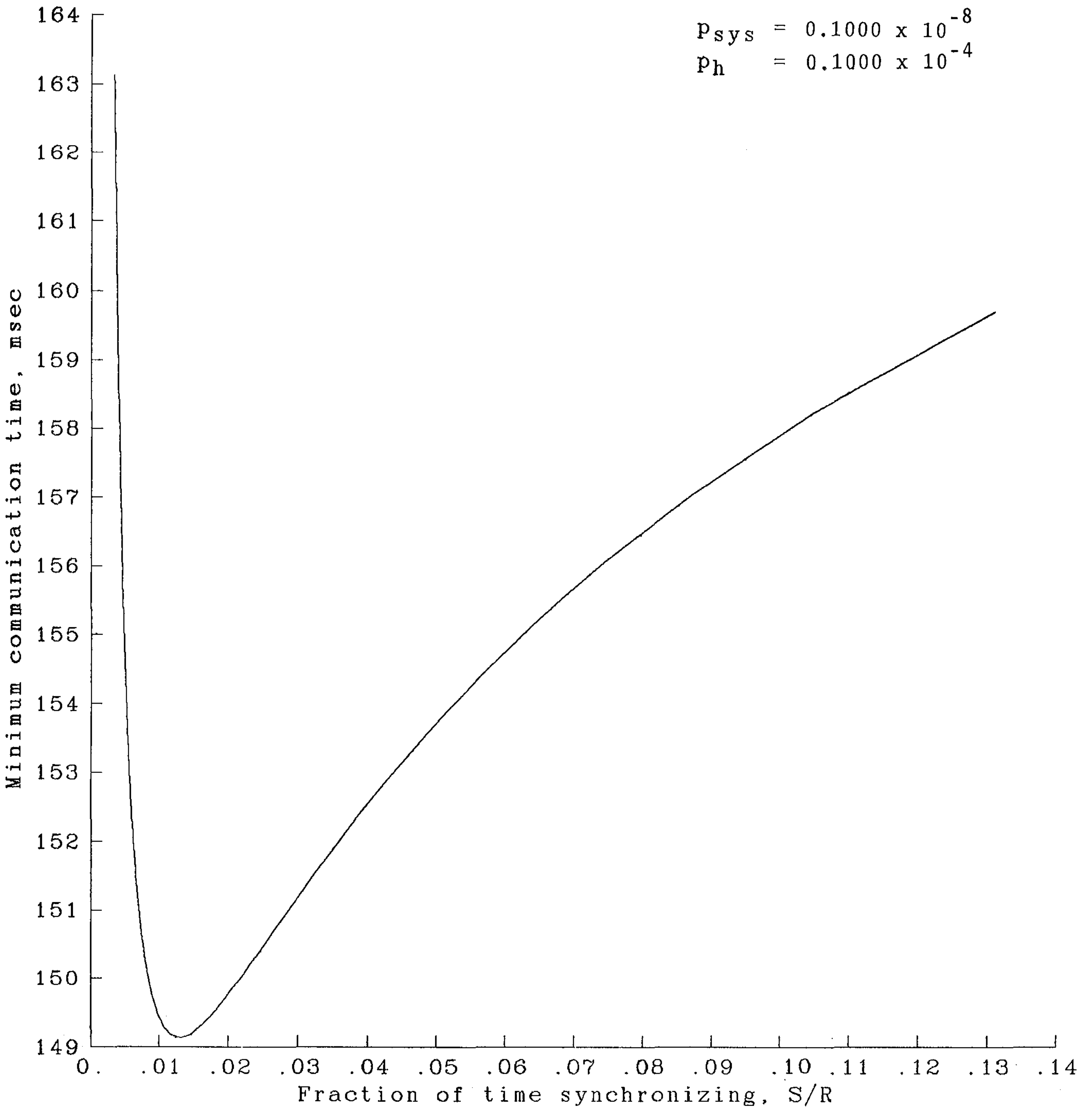

Figure 12. Performance-overhead trade-off.

Appendix

Derivation of a Simplified e_{qp} Formula

The synchronization theory is expressed in terms of real time. For example, the ϵ bound is defined in terms of the $r(T)$ function. Unfortunately, the state variables of a synchronization system are maintained in terms of clock times. In this section, the difference between the actual e_{qp} and one derived from clock measurements is shown to be negligible.

Lemma 1: If clock r is good, then $r(T_0+\Delta) \approx r(T_0)+\Delta$ for small Δ.

Proof: From the definition of a good clock,

$$\left|\frac{dr}{dT} - 1\right| < \frac{\rho}{2}$$

or

$$\left(1-\frac{\rho}{2}\right) < \frac{dr}{dT} < \left(1+\frac{\rho}{2}\right)$$

By integration,

$$\int_{T_0}^{T_0+\Delta}\left(1-\frac{\rho}{2}\right)dT < \int_{T_0}^{T_0+\Delta}\left(\frac{dr}{dT}\right)dT < \int_{T_0}^{T_0+\Delta}\left(1+\frac{\rho}{2}\right)dT$$

Evaluating these integrals yields

$$(1-\rho/2)\Delta < r(T_0+\Delta) - r(T_0) < (1+\rho/2)\Delta$$

or

$$|r(T_0+\Delta) - [r(T_0)+\Delta]| < (\rho/2)\Delta$$

Thus, if $(\rho/2)\Delta$ is negligible (ρ is typically on the order of 10^{-5}, and Δ is typically on the order of 10^{-6}), then

$$r(T_0+\Delta) \approx r(T_0)+\Delta$$

Lemma 2: If clock C is good, then $C(y+\Delta) \approx C(y)+\Delta$ for small Δ.

Proof: By the previous lemma, $r(T_0+\Delta) \approx r(T_0)+\Delta$. Letting $x = r(T_0+\Delta)$ and $y = r(T_0)$, then

$$C(x) = T_0+\Delta$$

$$C(y) = T_0$$

and

$$x \approx y+\Delta$$

Thus,

$$C(x)-\Delta = T_0 = C(y), \text{ and } C(x) \approx C(y+\Delta)$$

Hence,

$$C(y+\Delta) \approx C(y)+\Delta$$

Using these lemmas, it is easy to demonstrate that we can accurately determine e_{qp} from measured clock data:

Theorem: For good clocks p and q: $C_p(y)-C_q(y)+e_{qp} = \Delta_{qp}$.

Proof: By definition,

$$e_{qp} = r_p(T_0+\Delta_{qp}) - r_q(T_0)$$

Letting $x = r(T_0+\Delta_{qp})$ and $y = r_q(T_0)$ yields

$$C_p(x) - \Delta_{qp} = T_0 = C_q(y)$$

Since $e_{qp} = x - y$,

$$C_p(e_{qp}+y) - \Delta_{qp} = C_q(y)$$

From lemma 2, we conclude that

$$C_p(y) - C_q(y) + e_{qp} = \Delta_{qp}$$

References

1. Goldberg, Jack; Kautz, William H.; Melliar-Smith, P. Michael; Green, Milton W.; Levitt, Karl N.; Schwartz, Richard L.; and Weinstock, Charles B.: *Development and Analysis of the Software Implemented Fault-Tolerance (SIFT) Computer.* NASA CR-172146, 1984.
2. Weissman, Ishay: Estimation of Parameters and Large Quantiles Based on the k Largest Observations. *J. American Stat. Assoc.*, vol. 73, no. 364, Dec. 1978, pp. 812–815.
3. Pickands, James, III: Statistical Inference Using Extreme Order Statistics. *Ann. Stat.*, vol. 3, no. 1, Jan. 1975, pp. 119–131.
4. Lawless, J. F.: *Statistical Models and Methods for Lifetime Data.* John Wiley & Sons, Inc., c.1982.
5. Feller, William: *An Introduction to Probability Theory and Its Applications—Volume I*, Third ed. John Wiley & Sons, Inc., 1968, pp. 174–179.

1. Report No. NASA TP-2346	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle VALIDATION OF A FAULT-TOLERANT CLOCK SYNCHRONIZATION SYSTEM		5. Report Date September 1984
		6. Performing Organization Code 505-34-13-31
7. Author(s) Ricky W. Butler and Sally C. Johnson		8. Performing Organization Report No. L-15799
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665		10. Work Unit No.
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		13. Type of Report and Period Covered Technical Paper
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

This paper presents a new validation method for the synchronization susbystem of a fault-tolerant computer system. The method combines formal design verification with experimental testing. The design proof reduces the correctness of the clock synchronization system to the correctness of a set of axioms which are experimentally validated. Since the reliability requirements are often extreme, requiring the estimation of extremely large quantiles, an asymptotic approach to estimation in the tail of a distribution is employed.

17. Key Words (Suggested by Author(s)) Validation, Fault tolerance, Clock synchronization, Verification, Reliability analysis		18. Distribution Statement Unclassified—Unlimited; Subject Category 62	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 21	22. Price A02