1999

# Using Omissive Faults to Obtain Local Convergence in Partially Connected Networks

M. H. Azadmanesh
*University of Nebraska at Omaha*, azad@unomaha.edu

A. W. Krings
*University of Idaho*, krings@cs.uidaho.edu

1999. *Transactions of the Nebraska Academy of Sciences*, 25: 1–13

# USING OMISSIVE FAULTS TO OBTAIN LOCAL CONVERGENCE IN PARTIALLY CONNECTED NETWORKS

**M. H. Azadmanesh** and **A. W. Krings**

Computer Science Department, DSC 203
University of Nebraska at Omaha
Omaha, Nebraska 68182-0500
azad@unomaha.edu

Department of Computer Science
University of Idaho
Moscow, Idaho 83844-1010
krings@cs.uidaho.edu

This work was supported by NASA Nebraska Space Grant and EPSCoR programs.

*ABSTRACT*

*Approximate Agreement* is an important issue in fault-tolerant distributed computing where non-faulty processes exchange and vote upon their local values, to arrive at values which are within the range of the initial values of the non-faulty processes and within a predefined tolerance of each other. Results to date in Approximate Agreement, however, are not capable of exploiting omission faults. Omission faults are presumed not to occur or a predefined default value is substituted for those values not received, or they are globally discarded before the voting algorithm executes. As a result, hybrid fault models can not differentiate between *omissive* and *transmissive* faults.

The performance and fault tolerance expressions for completely connected networks, in the presence of omission faults, have recently been obtained. This paper develops a methodology which *logically* converts partially connected networks into completely connected networks. Hence, the results of completely connected systems can be applied to obtain the *local* convergence and fault tolerance expressions for partially connected systems.

† † †

Digital computers are essential to critical applications such as aerospace systems, air traffic control systems, nuclear power systems, computer manufacturing systems, etc. Common to all of these applications is the demand for maximum reliability and high performance from computer components. This requirement is necessarily stringent because a single component failure in these applications can lead to disaster. Because of such a stringent requirement, the fault-tolerant computing plays a significant part in the design of reliable and safe computers.

One way of making these applications ultra-dependable is to employ hardware/software redundancy, which brings into being many issues. One is synchronization and coordination among different computer components to achieve the expected services. The synchrony, in turn involves the creation of algorithms which ensure that the good components stay in synchrony in spite of faulty ones. For example, many applications in distributed systems require the clocks of processors to be synchronized so that the distributed events can be properly monitored and executed in the proper order. However, the clocks cannot stay in perfect harmony, as they cannot operate exactly at the same speed and the messages sent between processors incur uncertain delays. In such a situation, an *Approximate Agreement* algorithm can be used, where processors iteratively exchange their local clock values and vote until all non-faulty clocks converge into values within a prespecified range of each other. Agreement can easily be achieved if the system is fault-free, but it becomes very complex when faulty computers send wrong or even conflicting values to different computers. Formally, Approximate Agreement (Dolev et al. 1983, 1986) is defined by the following conditions:

A1: AGREEMENT — The voting algorithms executed by all non-faulty processes eventually halt with voted values that are within $\varepsilon$ of each other.

A2: VALIDITY — The voted value held by each non-faulty process is within the range of the initial values held by the non-faulty processes.

Many Approximate Agreement algorithms employ multiple rounds of message exchange. In each round, each process sends its value to all receiving processes. On receipt of a collection of values, each process executes an approximation function $F$ to obtain its latest voted value, which is used in the next round of message exchange. The objective of Approximate Agreement can be achieved by ensuring that each round is convergent, i.e. the range of the correct values is reduced in

each round. This property, called *single-step convergence*, guarantees that the range of values will eventually be less than $\varepsilon$, given enough rounds.

Section 2 gives the definitions for different failure modes. Section 3 describes the limitations of the existing voting algorithms and the motivation for this research. Section 4 introduces partially connected networks, and their impact on convergence properties. Section 5 describes the impact of omissive faults on voting algorithms. It also shows how a partially connected system can logically look like a completely connected network. Section 6 defines two sub-families of algorithms called dynamic-$\sigma$ and fixed-$\sigma$. Sections 7 and 8 show the convergence rate and fault tolerance for the two sub-families of algorithms. Section 9 provides an example to better understand the process of determining whether convergence is possible, using the expressions obtained in the previous sections. Finally Section 10 concludes the paper and comments on future research prospects.

## 2. FAULT MODE DEFINITIONS

Recent research has addressed convergent voting in the presence of multiple fault modes (Azadmanesh and Kieckhafer 1995, Kieckhafer and Azadmanesh 1993, 1994). This work uses the hybrid fault model of Thambidurai and Park (1988), which partitions faults into three modes: *benign*, *symmetric*, and *asymmetric*. Benign faults are defined as those which are self-incriminating or self-evident to *all* processes. A symmetric fault is defined as a fault whose value is perceived identically by all receiving non-faulty processes. An asymmetric fault is the one which is capable of sending conflicting (arbitrary) messages to different non-faulty processes. Using this hybrid fault model, the total number of faults, containing $a$ asymmetric, $s$ symmetric, and $b$ benign faults, is $t = a + s + b$. Under this fault model, simple expressions were derived for the performance and fault-tolerance of a broad family of convergent voting algorithms called Mean-Subsequence-Reduced (MSR) algorithms (Kieckhafer and Azadmanesh 1993, 1994).

Hybrid analysis of MSR produced more accurate bounds on the properties of the algorithms than possible with any single-mode fault model. However, these algorithms along with other traditional algorithms (Dolev et al. 1986, Kieckhafer and Azadmanesh 1994, Lamport and Melliar-Smith 1985, Meyer and Pradhan 1987, Thambidurai and Park 1988) *cannot* exploit the *omission* failure mode. An omission occurs when a process does not receive a value from a faulty process. These algorithms either assume that omissions do not occur or replace the omission with a predefined default value. However by a similar observation that Byzantine faults were partitioned into asymmetric and symmetric, asymmetric and symmetric faults can each be further subdivided into *transmissive* and *omissive* modes. A transmissive fault occurs when one or more processes receive erroneous values. An omissive fault occurs when a faulty process does not deliver its value to one or more processes. An asymmetric fault can be either transmissive, i.e. when a faulty process delivers conflicting values to all receiving processes, or it can be simultaneously transmissive and omissive, i.e. when a faulty process delivers a value to one or more processes and no value to others. On the other hand, symmetric faults, by definition, are either transmissive, i.e. the same erroneous value is delivered to all receiving processes, or are omissive when no value is delivered to any process.

Several failure modes can be classified under omissive faults, such as a *crash* fault or a *fail-stop* fault, where a process fails to transmit any messages, or a *timing* fault, where a process does not respond within the specified time frame (Cristian et al. 1985, 1986, 1989; Schneider 1984). In addition, by a modest amount of internal self-checking or using *authenticated* messages (Cristian et al. 1985, Wakerly 1978), the locally diagnosed benign errors can be transformed into omissive errors, increasing the count of the latter dramatically.

## 3. MOTIVATION

The existing voting algorithms can not take advantage of omissive faults because each process must deal with exactly the same number of messages in each round of voting. This number is fixed and is known *a priori*. This assumption creates the following negative consequences:

1. Omissive errors are transformed into more severe fault modes such as symmetric or asymmetric,
2. Locally diagnosed benign errors can not be discarded,
3. The voting algorithms become *less* fault-tolerant.

For completely connected systems, Azadmanesh and Kieckhafer (1996, 1998) have shown that the inclusion of omissive faults improves fault-tolerance, and that the need to globally diagnose benign errors is decreased, thus reducing the overhead of running a voting algorithm to recognize the global benign faults. Two subclasses of omissive faults were considered: strictly *omissive asymmetric* and *omissive symmetric*. A pro-

cess which behaves in a strictly omissive asymmetric manner sends the same "correct" value to some processes and no value to others, whereas, in omissive symmetric, the process value is received by no processes.

Our research will employ five types of failure modes: benign, transmissive symmetric, omissive symmetric, transmissive asymmetric, and strictly omissive asymmetric. Based on this fault model a new family of voting algorithms, called *Omission-MSR* will be introduced. The analysis will be done for synchronous systems (Dolev et al. 1983) with partial connectivity. The motivation for OMSR algorithms for partially connected systems is based on the following simple observations:

1. To treat omissive errors as omissive in order to improve fault-tolerance rather than converting them into more severe failure modes,

2. There are no general methods for synchronous, partially connected systems to measure the performance of different voting algorithms in the presence of omissive faults,

3. Omissive faults can be a predominant mode of failure in partially connected networks,

4. A partially connected system appears like a completely connected system with appropriate links behaving in omissive manner.

## 4. PARTIALLY CONNECTED SYSTEMS

The vast majority of research in convergent voting has considered only completely connected systems (Dolev et al. 1983, 1986; Kieckhafer and Azadmanesh 1994, Lamport and Melliar-Smith 1985, Vasanthavada and Marinos 1988, Vasanthavada and Thambidurai 1989). If the physical connectivity of the system is not complete, then it is assumed that messages are relayed by intervening processes to achieve complete "logical" connectivity. As a system grows large, so does the number of communication links, or the traffic required for message relays. Thus, the assumption of complete connectivity restricts the application of convergent voting to relatively small systems. In this research, however, *the relay of messages is prohibited*. As a result, each node receives only those messages initiated by its immediate neighbors. Global convergence must then occur with each process acting only on local information. This approach has the disadvantage that the system will converge more slowly than a completely connected system. However, it has the advantage that the overhead of messaging becomes independent of the number of nodes in the network.

While local convergence is a prerequisite to global convergence, it does not guarantee single step global convergence. Two immediate neighbors such as processes $i$ and $j$ may receive values from their respective neighbors that are not shared with each other. Since these values change with every round, processes $i$ and $j$ may diverge with respect to the previous round. Hence, during the course of global convergence a cluster of local nodes may go through a period of convergence and divergence before they finally converge. As a result, global convergence is *asymptotic* rather than monotonic (Kieckhafer and Azadmanesh 1993).

It is assumed that the system is a large, regular, sparsely connected network of $N$ processing nodes, each with degree $d$. The following describes the relationships between values received by two arbitrary non-faulty processes $i$ and $j$:

$\mathbf{P}_i$ = The set of processes adjacent to process $i$, including process $i$.

$\mathbf{P}_{i\cap j}$ = $\mathbf{P}_i \cap \mathbf{P}_j$, the set of processes adjacent to processes $i$ and $j$, including processes $i$ and $j$.

$\mathbf{P}_{i\cup j}$ = $\mathbf{P}_i \cup \mathbf{P}_j$, the set of processes adjacent to either or both of processes $i$ and $j$.

$\mathbf{U}_{i\cap j}$ = The multiset of *correct* values generated in $\mathbf{P}_{i\cap j}$.

$\mathbf{U}_{i\cup j}$ = The multiset of *correct* values generated in $\mathbf{P}_{i\cup j}$. Thus $\mathbf{U}_{i\cup j}$ is the multiset of correct values in $\mathbf{U}_{i\cap j}$ plus the multiset of values generated in $\mathbf{P}_i \backslash \mathbf{P}_j$ and $\mathbf{P}_j \backslash \mathbf{P}_i$.

$\chi$ = $|\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}| = |\mathbf{P}_j \backslash \mathbf{P}_{i\cap j}|$, the number of processes adjacent to $i$ or $j$ but not to both. In a completely connected system, since each node is adjacent to all nodes, $\chi = 0$.

$f$ = The maximum number of faulty processes in either $\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}$ or $\mathbf{P}_j \backslash \mathbf{P}_{i\cap j}$, regardless of their failure modes. Since these processes could behave omissively, $f_i \le f$, where $f_i$ is the number of erroneous values in $\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}$ received by $i$.

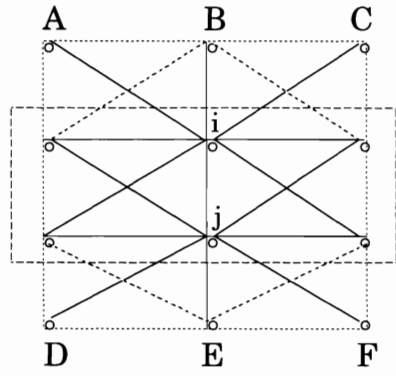


Figure 1. An Octagonal Mesh showing the link connections for nodes $i$ and $j$.

As an example, in Fig. 1, the solid lines represent the link connections for two processes $i$ and $j$ for a portion of a large octagonal mesh. $\chi$ is the number of processes in the top or the bottom row, so $\chi = 3$. In the top or bottom row, at most $f$ processes are assumed to be faulty. Thus, $f \leq \chi$. $\mathbf{P}_i$ consists of the top three rows of processes, while $\mathbf{P}_j$ consists of the bottom three rows. Thus, $\mathbf{P}_{i \cup j}$ consists of all nodes in the figure, while $\mathbf{P}_{i \cap j}$ consists of only those nodes within the dashed box.

In contrast to completely connected systems, where $\mathbf{U}_{i \cap j} = \mathbf{U}_{i \cup j}$, in partially connected systems $\mathbf{U}_{i \cap j} \subset \mathbf{U}_{i \cup j}$,. Therefore, convergence properties are impacted as to whether convergence is obtained with respect to the intersection or the union of the non-erroneous values for two arbitrary non-faulty processes. For instance, in Fig. 1, *Intersection Convergence* (IC) between processes $i$ and $j$ is obtained with respect to the values held by processes $\mathbf{P}_{i \cap j}$. Whereas, *Union Convergence* (UC) includes all the values held by processes shown in the figure, i.e. the values used in the IC plus those values in the top and the bottom rows. It will be shown that UC is less restrictive than IC simply because it will require less connectivity among processes. Furthermore, the effect of UC diffuses across the overlapping regions of local convergence faster than that of IC because it covers a larger subgraph than IC, i.e. $|\mathbf{U}_{i \cup j}| = |\mathbf{U}_{i \cap j}| + (\chi - f)$.

Given a voting algorithm $F(\mathbf{V})$, two processes $i$ and $j$ are Union Convergent if the following conditions are both true in every round of voting:

$F(\mathbf{V}_i) \in \rho(\mathbf{U}_{i \cup j})$, and $F(\mathbf{V}_j) \in \rho(\mathbf{U}_{i \cup j})$,

$|F(\mathbf{V}_i) - F(\mathbf{V}_j)| \leq C\delta(\mathbf{U}_{i \cup j})$, where $0 \leq C < 1$,

where:

$C$ = *Convergence rate*; it shows the effectiveness of a convergent voting algorithm.

$\mathbf{V}_i$ = $\langle v_{i,1},...v_{i,V_i} \rangle$; the multiset of real numbers received by process $i$ sorted such that $v_{i,k} \leq v_{i,k+1}\ \forall\ k \in \{1,...,V_i - 1\}$. $V_i$ is the size of $\mathbf{V}_i$.

$\rho(\mathbf{U}_{i \cup j})$ = $[\min(\mathbf{U}_{i \cup j}), \max(\mathbf{U}_{i \cup j})]$ $\rho(\mathbf{U}_{i \cup j})$ is called the *range* of $\mathbf{U}_{i \cup j}$.

$\delta(\mathbf{U}_{i \cup j})$ = $\max(\mathbf{U}_{i \cup j}) - \min(\mathbf{U}_{i \cup j})$. $\delta(\mathbf{U}_{i \cup j})$ is called the *diameter* of $\mathbf{U}_{i \cup j}$.

The conditions for IC are the same, except that $\mathbf{U}_{i \cup j}$ is replaced with $\mathbf{U}_{i \cap j}$. Another major difference between completely connected and partially connected systems is their handling of benign faults. We distinguish between *local* and *global* benign faults. A global benign error is recognized by all non-faulty processes in the system. In contrast, local benigns are recognized by only a subset of processes. In a completely connected system, global benign faults can be ignored because *all* processes can delete the benign errors from $\mathbf{V}$ and vote with a smaller sized multiset (Kieckhafer and Azadmanesh 1994). Thus, the multiset size $V$ is the same for all non-faulty processes. However, in a partially connected system without message relays, no value is received by all processes. Thus, *no fault is self-evident to all non-faulty processes* as required in the definition of a benign fault (Meyer and Pradhan 1987). Therefore, in partially connected systems, only symmetric and asymmetric faults are considered.

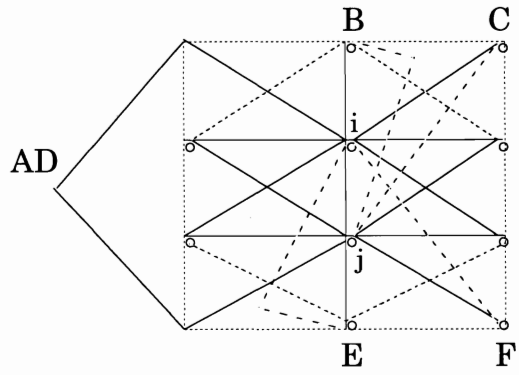


Figure 2. Completely connected view of Figure 1 with respect to nodes $i$ and $j$.

## 5. CONVERGENCE WITH OMISSION FAULTS

In previous studies of synchronous Approximate Agreement, $|\mathbf{V}_i| = |\mathbf{V}j| \ \forall\ i, j \in \{1,\ldots, N\}$. Thus, if an omissive error occurred, a default value was substituted for the missing value. Similarly, a process could not simply disregard its locally diagnosed transmissive errors. This is to ensure the equality of $|\mathbf{V}|$ for all non-faulty processes. As a result, the previous voting algorithms could not exploit the omissive behavior of malicious faults.

A variant of the MSR family (Kieckhafer and Azadmanesh 1994) of voting algorithms will be exploited. The new family of algorithms, called *Omission MSR*, or OMSR algorithms, differ from MSR algorithms in that the size of voting multiset $\mathbf{V}$ is no longer fixed and may change in every round of voting. During a voting round, omissive or self-evident transmissive errors are simply discarded. No defaults are substituted into the voting multiset $\mathbf{V}$. Thus, omissive errors remain omissive, and self-evident errors become omissive.

### 5.1. Conversion to complete connectivity

The voting algorithms for partially connected systems use the same approximation function as that used for completely connected systems (Azadmanesh and Kieckhafer 1997, Dolev et al. 1986, Kieckhafer and Azadmanesh 1994): $F(\mathbf{V}_i)$ = mean $[Sel_{\sigma_i}(Red^{\tau}(\mathbf{V}_i))]$. The "Reduction" function $Red^{\tau}$ removes the $\tau$ largest and $\tau$ smallest elements from multiset $\mathbf{V}_i$, in order to produce the *medial multiset* $\mathbf{M}_i$. The "Selection"" function $Sel_{\sigma_i}$ then selects $\sigma_i$ elements from $\mathbf{M}_i$, to produce the selected multiset $\mathbf{S}_i$. The final voted value $F(\mathbf{V}_i)$ is the arithmetic mean of the selected multiset.

With respect to two arbitrary neighbors such as $i$ and $j$, a partially connected system can look like a completely connected system. Figure 2 shows how the partially connected network in Figure 1 is converted to look like a completely connected network.

The network is completely connected because every node connected to $i$ is also connected to $j$. The dashed lines are logical rather than physical, so that any data sent on them are lost. In essence, these lines are the source for omissive faults. It is assumed that $A$ and $D$ are faulty, so $f = 1$. Since $A$ and $D$ each sends erroneous values to processes $i$ and $j$ respectively, they can be combined logically to form a single faulty node sending conflicting data to $i$ and $j$. This node is shown as $AD$. With respect to nodes $B$ and $C$, they are behaving in strictly omissive asymmetric manner. Their correct values are received by $i$ but not by $j$. A similar situation exists for nodes $E$ and $F$.

Azadmanesh and Kieckhafer (1996, 1997) have obtained the expressions for convergence rate and fault tolerance for completely connected systems. Therefore, by logically converting partial connectivity to complete connectivity, their results can be applied to partially connected networks. In Azadmanesh and Kieckhafer (1996, 1997), the parameters used to represent different fault modes are:

$\omega_{a_i}$ = The number of strictly omissive asymmetric values received by process $i$ but not received by process $j$. $\omega_a$ is the number of such faults in the network.

$a'_i$ = The number of transmissive asymmetric faults received by process $i$, i.e. the number of faults displaying any form of asymmetric behavior other than strictly omissive. $a'$ is the number of such faults in the network.

$s'$ = The number of transmissive symmetric faults in the network.

$\omega_s$ = The number of omissive symmetric faults in the network.

$b$ = The number of benign faults in the network.

Let $t_{complete}$ be the total number of faults in a truly completely connected network. Then:

$$t_{complete} = (a' + \omega_a) + (s' + \omega_s) + b.$$

The relationships between this new and the previous fault partitionings (Dolev et al. 1986, Kieckhafer and Azadmanesh 1994, Thambidurai and Park, 1988) which cannot exploit omissive faults are specified by the relations: $a = a' + \omega_a$ and $s = s' + \omega_s$ .

As indicated, benign faults are not applicable to partially connected systems, so $b = 0$. For the other fault modes, by closely looking at Figure 2:

$$a' = a_{i\cap j} + f \tag{5.1}$$

$$\omega_a = \omega_{a_{i\cap j}} + 2(\chi - f) \tag{5.2}$$

$$s' = s_{i\cap j} \tag{5.3}$$

$$\omega_s = \omega_{s_{i\cap j}} \tag{5.4}$$

where:

$a_{i\cap j}$ = The number of transmissive asymmetric faults in $\mathbf{P}_{i\cap j}$.

$\omega_{a_{i\cap j}}$ = The number of strictly omissive asymmetric faults in $\mathbf{P}_{i\cap j}$.

$s_{i\cap j}$ = The number of transmissive symmetric faults in $\mathbf{P}_{i\cap j}$.

$\omega_{s_{i\cap j}}$ = The number of omissive symmetric faults in $\mathbf{P}_{i\cap j}$.

In (5.2), $2(\chi - f)$ is justified because processes $i$ and $j$ each does not receive $(\chi - f)$ of the values received by the other process. Let $t_{partial}$ be the total number of faults in the partially connected network. Then, replacing the parameters in $t_{complete}$ with the expressions in (5.1) – (5.4) yields:

$$t_{partial} = (a_{i\cap j} + \omega_{a_{i\cap j}}) + (s_{i\cap j} + \omega_{s_{i\cap j}}) + 2\chi - f$$

Note that $t_{partial}$ is the total number of local faults with respect to two arbitrary processes $i$ and $j$, and not the total number of faults in the system.

In completely connected networks, benign faults can be discarded a priori before the voting algorithm executes, because they are globally diagnosed and thus every process is aware of them. Furthermore, a process does not receive values from those processes which behave in symmetric omissive manner, i.e. $\omega_s$ . Thus, in a truly completely connected network, the total number of values received by a process $i$ from faulty processes is:

$$t_{i,complete} = (a'_i + \omega_{a_i}) + s'$$

Mapping this equation to Figure 2 yields:

$$a_i' = a_{i,i\cap j} + f_i \tag{5.5}$$

$$\omega_{ai} = \omega_{ai,i\cap j} + (\chi - f_i) \tag{5.6}$$

$$s' = s_{i\cap j} \tag{5.7}$$

where $a_{i,i\cap j}$ is the number of transmissive asymmetric faults in $\mathbf{P}_{i\cap j}$ received by process $i$, and $\omega_{a_{i,i\cap j}}$ is the number of strictly omissive asymmetric faults in $\mathbf{P}_{i\cap j}$ received by process $i$. Applying the expressions in (5.5) – (5.7) to $t_{i,complete}$ , shows that, in a partially connected network, the total number of values received by process $i$ from faulty processes is:

$$\begin{aligned} t_{i,partial} &= (a_{i,i\cap j} + f_i + \omega_{ai,i\cap j} + (\chi - f_i)) + s_{i\cap j} \\ &= a_{i,i\cap j} + \omega_{ai,i\cap j} + s_{i\cap j} + \chi \end{aligned}$$

Herein, an "error" is defined as any value received from a faulty process that does not behave in strictly omissive asymmetric manner because, even though, a strictly omissive asymmetric process is faulty, its values are "correct." Hence, the maximum number of erroneous values received by any non-faulty process is $(a' + s' + f)$. As a result, $\tau \geq (a' + s' + f)$ ensures that $\rho(Red^{\tau}(\mathbf{V}_i)) \subseteq \rho(\mathbf{U}_{i\cup j})$, so that $F(\mathbf{V})$ generates a value within the range of correct values.

With respect to the total number of faults in a partially connected network, two alternatives can be identified. One is to place a limit on the number of faults in the entire system, as done in Ramanathan et al., 1990. However, in a large distributed system, it may not be practical to place such a limit. The other approach is to place a limit on the number of faults received by a process, as done in this paper. This approach is more realistic but has the disadvantage that if a non-faulty process becomes divergent, due to the diffusion of local faults into other areas of the system, the entire system may become divergent. The example at the end of the paper sheds more light on this issue.

### 5.2 Definition of $\gamma$

Let $s_{i,g}$ be any element of the selected multiset $\mathbf{S}_i$, and let $m_{i,k_i(g)}$ be the corresponding element in the medial multiset $\mathbf{M}_i$. Then, for each g $\in \{1,...,\sigma_i\}$ there exists exactly one $k_i(g) \in \{1,...,|\mathbf{M}_i|\}$ which guarantees that $s_{i,g} = m_{i,k_i(g)}$ for all possible $\mathbf{M}_i$. Given two indices into $\mathbf{S}_i, g$ and $h \in \{1,...,\sigma_i\}$, where $g \leq h$, define $\Delta k_i(g,h) = k_i(h) - k_i(g)$ as the number of elements in $\mathbf{M}_i$ *spanned* by elements $(s_{i,g},...,s_{i,h})$ in $\mathbf{S}_i$.

Now, define the parameter $\gamma_{i,z}$ as the *minimum* value which ensures that $\Delta k_i((g, g+\gamma_{i,z}) \geq z$, for all $g \in \{1,...,\sigma_i - \gamma_{i,z}\}$. By this definition, $\gamma_{i,z}$ exists only if $|\mathbf{M}_i| > z$. Furthermore, if $\gamma_{i,z}$ exists then $\gamma_{i,z} \leq \sigma_i$. It will be shown that the expression for convergence rate will depend on this parameter.

## 6. FIXED AND DYNAMIC VOTING ALGORITHMS

As indicated, in OMSR, the size of $\mathbf{V}$ for each process can be different. This implies that medial multisets might be of different sizes. Accordingly, depending on the selection function, each processing node may deal with a different number of selected elements. This, and the fact that not all selection functions belong to the same family of algorithms, created the need to distinguish among different families of algorithms. There are two general families of selection functions: Fixed-$\sigma$ and Dynamic-$\sigma$. A Fixed-$\sigma$ selection function always selects the same number of entries from the multiset $\mathbf{M}$, regardless of the size of $\mathbf{M}$. By contrast, in a Dynamic-$\sigma$ selection function, the number of entries selected from $\mathbf{M}$ depends on the size of $\mathbf{M}$, i.e. $\sigma$ is a function of $|\mathbf{M}|$. Each family of algorithms could contain many sub-families. Two sub-families, one from each family, are considered in such a way to ensure that together they encompass all commonly used voting algorithms (Dolev et al. 1983, Kieckhafer and Azadmanesh 1994).

### 6.1 Dynamic-$\sigma$ selection functions

The study of Dynamic-$\sigma$ selection functions is limited to the sub-family of *enumerative* selection functions. Define the medial multiset $\mathbf{M}_{max} = \left\langle m_1,\ldots,m_{|\mathbf{M}_{max}|} \right\rangle$, and $\mathbf{S}_{max} = Sel_{\sigma_{max}}(\mathbf{M}_{max})$, where $\sigma_{max}$ is the number of elements selected from $\mathbf{M}_{max}$. Also, define the enumerative selection set as a set of integers $E = \{e_1,\ldots,e_{\sigma_{max}}\}$ whose elements are the indices of all elements of $\mathbf{M}_{max}$, where $e_j < e_{j+1}\ \forall j \in \{1,\ldots,\sigma_{max}-1\}$. For any process $i$, the selected multiset $\mathbf{S}_i$ is then: $Sel_{\sigma_i}(\mathbf{M}_i) = \langle m_{i,e_1},\ldots,m_{i,e_{\sigma_i}}\rangle$, where $\sigma_i$ is the largest value such that $e_{\sigma_i} \le |\mathbf{M}_i|$. In other words, the elements selected from $\mathbf{M}_i$ are those whose indices appear in $E$.

### 6.2 Fixed-$\sigma$ selection functions

For this family of algorithms, $\sigma_i = \sigma_j$, for any pair of $\mathbf{M}_i$ and $\mathbf{M}_j$ Hence, the convergence rate expression becomes simpler because $\sigma_{max} = \sigma_{min}$. The subfamily of selection functions adopted has the following properties:

$$|\mathbf{M}_i| \ge |\mathbf{M}_j| \Rightarrow \begin{cases} k_i(g) \ge k_j(g), & \forall g \in \{1,\ldots,\sigma\} \\ \Delta k_i(g,g+1) \ge \Delta k_j(g,g+1), & \forall g \in \{1,\ldots,\sigma-1\} \end{cases}$$

Informally, these properties state that as $|\mathbf{M}|$ increases the number of elements between each pair of selected elements and the index of any selected element in $\mathbf{M}$ does not decrease.

## 7. DYNAMIC-$\sigma$ CONVERGENCE RATE

### 7.1 Union convergence

**Theorem 1**: *Given an enumerative dynamic-$\sigma$ selection function, and two multisets $\mathbf{V}_i$ and $\mathbf{V}_j$, such that $F(\mathbf{V}_i) \ge F(\mathbf{V}_j)$, the UC rate is:*

$$C \le \begin{cases} \dfrac{\gamma_{j,a_{j,i\cap j}} + \omega_{a_{j,i\cap j}} + \chi}{\sigma_j} & : \ \sigma_i \le \sigma_j \\ \dfrac{\sigma_i - \sigma_j + \gamma_{j,a_{j,i\cap j}} + \omega_{a_{j,i\cap j}} + \chi}{\sigma_i} & : \ \sigma_i \ge \sigma_j \end{cases} \tag{7.1}$$

**Proof**: It has been shown (Azadmanesh and Kieckhafer 1998) that the convergence rate for a completely connected network is:

$$C \le \begin{cases} \dfrac{\gamma_{j,a'_j} + \omega_{a_j}}{\sigma_j} & : \ \sigma_i \le \sigma_j \\ \dfrac{\sigma_i - \sigma_j + \gamma_{j,a'_j} + \omega_{a_j}}{\sigma_i} & : \ \sigma_i \ge \sigma_j \end{cases}$$

Since we indicated that a partially connected system, from the perspective of two non-faulty nodes $i$ and $j$, can be viewed like a completely connected systems, where each missing link is behaving like an omissive fault, we can safely use (5.5) and (5.6). Thus, by replacing $a'_j$ with $a_{j,i\cap j} + f_j$ and $\omega_{a_j}$ with $\omega_{a_{j,i\cap j}} + (\chi - f_j)$, the convergence rate in (7.1) is obtained. □

**Theorem 2**: *Given an enumerative dynamic-$\sigma$ selection function, and two multisets $\mathbf{V}_i$ and $\mathbf{V}_j$, such that $F(\mathbf{V}_i) \ge F(\mathbf{V}_j)$, the voting algorithm can be convergent only if:*

$$V_j \ge 2\tau + \max(a_{j,i\cap j} + \omega_{a_{j,i\cap j}} + \chi + 1, \sigma_j), \quad \tau \ge a_{i\cap j} + s_{i\cap j} + f \tag{7.2}$$

**Proof**: For a completely connected system, it has been shown that (Azadmanesh and Kieckhafer 1998):

$$V_j \geq 2\tau + \max(a'_j + \omega_{a_j} + 1, \sigma_j), \quad \tau \geq a' + s' \tag{7.3}$$

Applying the expressions in (5.5) – (5.7) and (5.1) to (7.3) yields:

$$\begin{aligned} V_j &\geq 2\tau + \max(a_{j,i\cap j} + f_j + \omega_{a_{j,i\cap j}} + (\chi - f_j) + 1, \sigma_j), \quad \tau \geq (a_{i\cap j} + f) + s_{i\cap j} \\ &= 2\tau + \max(a_{j,i\cap j} + \omega_{a_{j,i\cap j}} + \chi - 1, \sigma_j), \quad \tau \geq a_{i\cap j} + s_{i\cap j} + f \end{aligned}$$ □

**Theorem 3**: *Given an enumerative dynamic-σ selection, a Union convergent voting algorithm exists if the following is true:*

$$\mathbf{P}_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1 \tag{7.4}$$

**Proof**: Let the number of nodes in Fig. 1 be $N_{partial}$. When this figure is converted to the completely connected form, i.e. Fig. 2, the number of nodes is $N_{complete} = N_{partial} - f$. It has been shown (Azadmanesh and Kieckhafer 1998) that a completely connected network with $N_{complete}$ nodes must satisfy the following inequality for the algorithm to be convergent: $N_{complete} \geq 3a' + 2s' + \omega_a + \omega_s + b + 1$. Using equations (5.1) – (5.4), and the facts that $b = 0$ and $N_{complete} = N_{partial} - f$, changes the inequality to:

$$N_{partial} - f \geq 3(a_{i\cap j} + f) + 2s_{i\cap j} + \left[\omega_{a_{i\cap j}} + 2(\chi - f)\right] + \omega_{s_{i\cap j}} + 1$$

After simplification, we get:

$$N_{partial} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2(\chi + f) + 1 \tag{7.5}$$

Now, since $\mathbf{P}_{i\cap j} = N_{partial} - 2\chi$, (7.5) becomes:

$$\mathbf{P}_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1$$ □

MSR algorithms in partially connected systems do not distinguish between transmissive and omissive faults. As a result, all strictly omissive asymmetric and omissive symmetric faults are treated as transmissive faults. If we use the notation $a_{a_{i\cap j}} + \omega_{a_{i\cap j}}$ and $s_{s_{i\cap j}} + \omega_{s_{i\cap j}}$ to represent all asymmetric and symmetric faults within $\mathbf{P}_{i\cap j}$ respectively, Kieckhafer and Azadmanesh (1993) have shown that a convergent voting algorithm exists if:

$$\mathbf{P}_{i\cap j} \geq 3a_{a_{i\cap j} + \omega_{a_{i\cap j}}} + 2s_{s_{i\cap j} + \omega_{s_{i\cap j}}} + 2f + 1 \tag{7.6}$$

Applying the same notation to the result of Theorem 3 yields:

$$\begin{aligned} \mathbf{P}_{i\cap j} &\geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1 \\ &= 3(a_{i\cap j} + \omega_{a_{i\cap j}}) + 2(s_{i\cap j} + \omega_{s_{i\cap j}}) + 2f + 1 - (2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}) \\ &= 3a_{a_{i\cap j} + \omega_{a_{i\cap j}}} + 2s_{s_{i\cap j} + \omega_{s_{i\cap j}}} + 2f + 1 - (2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}) \end{aligned} \tag{7.7}$$

Comparing (7.7) to (7.6), it is observed that including omissive faults into a fault model reduces $\mathbf{P}_{i\cap j}$ by $\left(2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}\right)$. Since omissions can be the dominant mode of failure in large geographically distributed networks,

saving $\left(2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}\right)$ nodes can be very significant.

Another advantage of OMSR over other voting algorithms, including MSR algorithms, is that self-evident errors need to be diagnosed only by the local processes. A process, upon the detection of an error, can simply drop the erroneous value from its voting multiset **V**. If the error is detected by every process, the effect will be the same as a globally diagnosed benign error. However, if the error is detected by just a subset of processes, then diagnosis at the global level will be of no help. As a result, employing OMSR algorithms reduces the need to globally diagnose errors.

### 7.2 Intersection convergence

IC requires a more restrictive criterion than UC. Specifically, $F(\mathbf{V})$ must be within the range $\mathbf{U}_{i\cap j}$. Therefore, the $(\chi - f)$ correct nodes in $\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}$ and $\mathbf{P}_j \backslash \mathbf{P}_{i\cap j}$ must be treated as faulty, regardless of their health, and thus $\tau$ must account for these nodes to ensure that the nodes in $\mathbf{P}_{i\cap j}$, after applying the voting algorithm, will generate values in $\rho\left(\mathbf{U}_{i\cap j}\right)$. Accordingly, $\tau \geq a_{i\cap j} + s_{i\cap j} + \chi$.

The conditions to ensure two processes $i$ and $j$ are Intersection convergent can be derived as a variant on UC described previously. In Theorems 1–3, the results are valid for any $f \leq \chi$. By setting $f = \chi$, it follows that $\rho(\mathbf{U}_{i\cap j}) \equiv \rho(\mathbf{U}_{i\cup j})$. Thus, the convergence rate and fault-tolerance expressions for IC are the same as those of UC except that $f$ is replaced with $\chi$.

It should be noted that, although the expressions for UC and IC rates are the same, they may not produce the same results because $\tau$ for the two environments are different. This will affect the medial multisets and in turn different $\sigma$ values will be produced for each environment.

## 8. FIXED-$\sigma$ SELECTION FUNCTIONS

In fixed-$\sigma$, the number of elements selected is fixed, regardless of the size of **V**. The method to obtain the expressions for convergence rate and fault-tolerance is the same as that of dynamic-$\sigma$, except that for any $\ell^{\text{th}}$ selected element, the equality $k_i(\ell) = k_j(\ell)$ may no longer be true. The next two theorems obtain these expressions only for UC. As in the dynamic-$\sigma$ case, the results for IC are the same except that $f$ is replaced with $\chi$.

**Theorem 4**: *Given a fixed-$\sigma$ selection function, the UC rate is:*

$$C = \frac{\gamma_{a_{i\cap j} + \omega_{a_{i\cap j}}} + \chi}{\sigma} \tag{8.1}$$

**Proof**: For a completely connected system, it has been shown (Azadmanesh and Kieckhafer 1998) that:

$$C = \frac{\gamma_\alpha}{\sigma} \tag{8.2}$$

where $\alpha$ represents the maximum effective number of asymmetric values seen by any non-faulty process. Consider two processes $i$ and $j$ as in Fig.1. The maximum effective number of asymmetric values seen by a process $i$ in $\mathbf{P}_{i\cap j}$ is $a_{i\cap j} + \omega_{a_{i\cap j}}$. The maximum number of asymmetric values seen by the same process from those processes whose values are not received by process $j$ is $\chi$. Thus, the total number of effective asymmetric values is $(a_{i\cap j} + \omega_{a_{i\cap j}} + \chi)$. By replacing this expression for $\alpha$ in (8.2), (8.1) is obtained. □

**Theorem 5**: *Given a fixed-$\sigma$ selection function, a Union convergent voting algorithm exists if the following is true*:

$$\mathbf{P}_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1$$

**Proof**: The proof is similar to Theorem 3. □

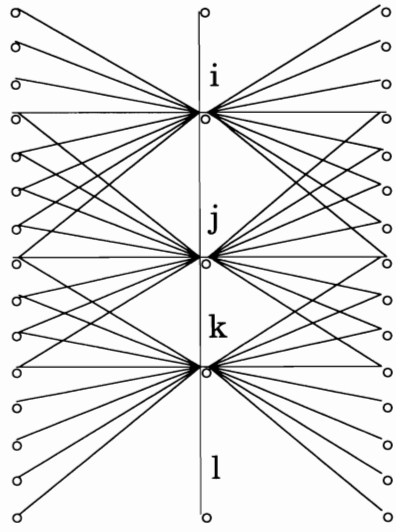


Figure 3. A partially connected network.

## 9. APPLICATION EXAMPLE

This example uses different values for the fault modes to show when UC is possible. A portion of a regular network is shown in Fig. 3. In this network, four nodes are labeled $i, j, k$, and $l$. The link connections are shown only for the nodes $i, j$, and $k$. For readability sake, let us use the notation $\chi_{i,j}$ to represent the number of periphery processes in $\mathbf{P}_i \setminus \mathbf{P}_{i \cap j}$, and use $f_{i,j}$ to show the faulty processes in $\mathbf{P}_i \setminus \mathbf{P}_{i \cap j}$. By inspection, $\mathbf{P}_{i \cap j} = 12$, $\mathbf{P}_{j \cap k} = 10$, $\chi_{i,j} = 7$, and $\chi_{j,k} = 9$. According to Theorem 3, $i$ and $j$ are convergent if: $\mathbf{P}_{i \cap j} \geq 3a_{i \cap j} + 2s_{i \cap j} + \omega_{a_{i \cap j}} + \omega_{s_{i \cap j}} + f_{i,j} + 1$, and $j$ and $k$ are convergent if $\mathbf{P}_{j \cap k} \geq 3a_{j \cap k} + 2s_{j \cap k} + \omega_{a_{j \cap k}} + \omega_{s_{j \cap k}} + f_{j,k} + 1$. Thus, convergence between $i$ and $j$, and between $j$ and $k$ is possible if:

$$12 \geq 3a_{i \cap j} + 2s_{i \cap j} + \omega_{a_{i \cap j}} + \omega_{s_{i \cap j}} + 2f_{i,j} + 1 \tag{9.1}$$

$$10 \geq 3a_{j \cap k} + 2s_{j \cap k} + \omega_{a_{j \cap k}} + \omega_{s_{j \cap k}} + 2f_{j,k} + 1 \tag{9.2}$$

Using these conditions, the examples in Table 1 show whether convergence exists between $i$ and $j$, and between $j$ and $k$. While a pair of nodes is convergent with respect to each other's correct values, a different pair of nodes may not be convergent. For instance, in row 2 of the table, processes $i$ and $j$ are convergent, whereas $j$ and $k$ are not. This peculiarity does not exist with completely connected systems, i.e. if a pair of nodes are convergent, every pair in the network is convergent. This, however, is not necessarily an indication that the entire network is divergent. As long as $\tau \geq a' + s' + f$, voted values will be within the range of the correct values. As the voted values of the convergent nodes diffuse across the network, they will eventually force the divergent nodes to become convergent. As a result, nodes may be divergent with respect to each other for more than a round of voting before reaching the point where the range of the voted values is smaller than the range of the correct values in the entire network.

In row 4, there are 3 transmissive asymmetric faults in $\mathbf{P}_{i \cap j}$ and also 3 such faults exist in $\mathbf{P}_{j \cap k}$. To make the situation worse, assume there are 3 transmissive asymmetric faults in $\mathbf{P}_{k \cap l}$. For this distribution of faults, the pairs $(i, j)$, $(j, k)$, and $(k, l)$ are all convergent. This accounts for 9 faulty processes in the network. However, if any process encounters that many faults, i.e. $a_{i \cap j} = 9$, there is no guarantee for the process to produce values within the range of the correct values. Once a non-faulty process becomes divergent, it may never again generate values within the range of the initial correct values. The process would then act exactly like a faulty process, even though it is not faulty. It may thus infect other processes which in turn may infect others. As a result, the entire network may never become convergent. The point is that global convergence is very dependent upon the distribution of faults within the network. One solution is to set a limit on the number of faults in the network while making sure that every process stays convergent within this limit of faults by adjusting $\tau$ accordingly. This is however

Table 1. Examples of convergence between processes $(i,j)$, and between $(j,k)$.

| Example # | $a_{i\cap j}$ and $a_{j\cap k}$ | $s_{i\cap j}$ and $s_{j\cap k}$ | $\omega_{a_{i\cap j}}$ and $\omega_{a_{j\cap k}}$ | $\omega_{s_{i\cap j}}$ and $\omega_{s_{j\cap k}}$ | $f_{i,j}$ and $f_{j,k}$ | (9.1) true? | (9.2) true? |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 0 | 2 | 1 | Yes | No |
| 2 | 0 | 3 | 0 | 3 | 1 | Yes | No |
| 3 | 0 | 0 | 0 | 9 | 0 | Yes | Yes |
| 4 | 3 | 0 | 0 | 0 | 0 | Yes | Yes |
| 5 | 3 | 0 | 1 | 1 | 0 | Yes | No |
| 6 | 2 | 3 | 0 | 0 | 0 | No | No |

restrictive because in a large network the limit will then be determined by the density of the connections between each pair of processes rather than the number of nodes in the network.

## 10. CONCLUSIONS

This paper considered two subfamilies of algorithms called dynamic-$\sigma$ and fixed-$\sigma$. Since in OMSR, voting multisets are no longer of the same size, and the fact that not all common voting algorithms belong to the same family of algorithms, it was not possible to employ a single family of algorithms encompassing all such algorithms. For instance, Fault-Tolerant Midpoint belongs to fixed-$\sigma$ but not to dynamic-$\sigma$, or Fault-Tolerant Mean belongs to dynamic-$\sigma$ but not to fixed-$\sigma$. Using these two subfamilies, it was shown that the inclusion of omissive faults improves fault tolerance in comparison to other models such as MSR.

Obtaining convergence rate directly from partial connectivity is a very tedious process (Azadmanesh and Kieckhafer 1995b) because the voting multisets are not of equal sizes, which introduces a number of difficulties. But it was shown that, from the perspective of two adjacent non-faulty nodes, partially connected systems can be viewed like completely connected systems. This made obtaining the expressions for convergence rate and fault tolerance manageable. It is conjectured that the methodology developed herein can be extended to any two nodes

Table 2. Summary of necessary dynamic-$\sigma$ and fixed-$\sigma$ convergence parameters.

| **Union** | **Intersection** |
|---|---|
| $\tau \geq a_{i\cap j} + s_{i\cap j} + f$ | $\tau = a_{i\cap j} + s_{i\cap j} + \chi$ |
| $P_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1$ | $P_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2\chi + 1$ |
| dynamic-$\sigma$: $C \leq \left[ \frac{\gamma_{j,a_{j,i\cap j}+\omega_{a_{j,i\cap j}}+\chi}}{\sigma_j} \;:\; \sigma_i \leq \sigma_j,\ \frac{\sigma_i - \sigma_j + \gamma_{j,a_{j,i\cap j}+\omega_{a_{j,i\cap j}}+\chi}}{\sigma_i} \;:\; \sigma_i \geq \sigma_j \right]$ | |
| fixed-$\sigma$: $C \leq \frac{\gamma_{a_{i\cap j}+\omega_{a_{i\cap j}}+\chi}}{\sigma}$ | |

in the network. This is already under investigation. If this conjecture is true, then the performance of global convergence can be obtained.

Table 2 summarizes the convergence bounds for fixed-$\sigma$ and dynamic-$\sigma$ selection functions under the OMSR fault-model. In this table, if omissive faults are presumed not to occur or are treated as parts of asymmetric and symmetric faults, then OMSR model will converge into the three-mode MSR fault model, because then all multisets will be of the same size.

## LITERATURE CITED


Azadmanesh, M. H., and R. W. Kieckhafer. 1995a. The general convergence problem: A unification of synchronous and asynchronous systems. *Dependable Computing and Fault-Tolerant Systems* 9: 251–267.

———, and ———. 1995b. Hybrid Fault Mode Analyses in Partially Connected Networks. University of Nebraska at Omaha, Computer Science Department, Technical Report No. 2: 42 pp.

———, and ———. 1996. New hybrid fault models for asynchronous approximate agreement. *IEEE Trans. on Computers* 45(4): 439–449.

———, and ———. 1998. Exploiting omissive faults in synchronous approximate agreement. *IEEE Trans. on Computers,* in review.

Cristian, F., H. Aghili, and R. Strong. 1986. Clock synchronization in the presence of omission and performance faults, and processor joins. *Sixteenth International Conference on Fault-Tolerant Computing.*

———, ———, ———, and D. Dolev. 1985. Atomic broadcast: from simple message diffusion to Byzantine agreement. *Proc. Fifteenth International Symposium on Fault-Tolerant Computing*: 200–206.

———, D. Dolev, R. Strong, and H. Aghili. 1989. Atomic broadcast in a real-time environment. Almaden Research Center, IBM Research Report.

Dolev, D., N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl. 1983. Reaching approximate agreement in the presence of faults. *Proc. Third Symposium on Reliability in Distributed Software and Database Systems.*

———, ———, ———, ———, and ———. 1986. Reaching approximate agreement in the presence of faults. *JACM* 33(3): 499–516.

Kieckhafer, R. M., and M. H. Azadmanesh. 1993. Low cost approximate agreement in partially connected networks. *Journal of Computing and Information* 3(2): 53–85.

———, and ———. 1994. Reaching approximate agreement with mixed mode faults. *IEEE Trans. on Parallel and Distributed Systems* 5(1): 53–63.

Lamport, L., and P. M. Melliar-Smith. 1985. Synchronizing clocks in the presence of faults. *JACM* 32(1): 52–78.

Meyer, F. J., and D. K. Pradhan. 1987. Consensus with dual failure modes. *Proc. Seventeenth Fault-tolerant Computing Symposium*: 48–54.

Ramanathan, P., et al. 1990. Hardware-assisted software clock synchronization for homogeneous distributed systems. *IEEE Trans. on Computers* C-39(4): 514–524.

Schneider, F.B. 1984. Byzantine generals in action: Implementing fail-stop processors. *ACM Trans. on Computer Systems* 2(2): 145–154.

Thambidurai, P. M., and Y. K. Park. 1988. Interactive consistency with multiple failure modes. *Proc. Seventh Reliable Distributed Systems Symposium.*

Vasanthavada, N., and P.N. Marinos. 1988. Synchronization of fault-tolerant clocks in the presence of malicious failures. *IEEE Trans. on Computers* C-37(4): 440–448.

———, and P. Thambidurai. 1989. Design of fault-tolerant clocks with realistic assumptions. *Proc. Eighteenth Fault-Tolerant Computing Symposium*: 128–133.

Wakerly, J. 1978. *Error Detecting Codes, Self-Checking Circuits and Applications.* New York, North Holland.