UNO-CS-TR-97-1

A Step toward Global Convergence in Partially Connected Networks

by

M.H. Azadmanesh and A.W. Krings

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF NEBRASKA AT OMAHA

A Step toward Global Convergence in Partially Connected Networks[1]

M.H. Azadmanesh [2] A. W. Krings [3]

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 priori* default value is substituted for those values not received, or they are globally discarded from the voting multisets 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. This way the results of completely connected systems can be applied to obtain the *local* convergence and fault tolerance expressions for partially connected systems. This methodology provides a foundation for on-going research in *global* convergence for partially connected networks.

Index Terms— Approximate Agreement, Byzantine Agreement, Clock Synchronization, Convergent Voting Algorithms, Fault-Tolerant Multiprocessors, Hybrid Faults.

---

[1] This work was supported by NASA Space Grant Consortium

[2] Department of Computer Science, University of Nebraska at Omaha, 68182–0459. E-mail: azad@ahvaz.unomaha.edu, Phone: 402-554-3976, Fax: 402-554-3747

[3] Department of Computer Science, Moscow, ID 83844-1010. E-mail: krings@cs.uidaho.edu, Phone: 208 885-4078, Fax: 208 885-9052

1 Introduction

With the proliferation of distributed systems, the *Agreement problem* has become one of the fundamental issues in reliable distributed computing. The basic goal is for each correct process to reach a consensus based on the values received from other processes. Agreement can easily be achieved if the system is fault-free. But it becomes very complex when faulty processes send wrong values or even conflicting values to different processes.

The Agreement can be *Exact* or *Inexact*. In Exact Agreement [12], the correct processes must agree on a binary value. It is required that all processes agree on the same value. Whereas, in Inexact Agreement, each correct process holds an estimate of the true value within a predefined tolerance ϵ [7, 8]. Thus, each process agrees on a real value rather than a discrete value. Formally, Approximate Agreement 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 ϵ 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 non-faulty values is reduced in each round. This property, called *single-step convergence*, guarantees that the range of values will eventually be less than ϵ, given enough rounds.

In general, there are two forms of voting algorithms: *synchronous* and *asynchronous* [7]. In a synchronous distributed system the maximum processing and the communication delays of non-faulty processes are bounded. Any data arriving after this defined time bound is considered to be from a faulty process. By contrast, asynchronous systems impose no bounds on process operation [7]. It is thus impossible to differentiate between a slow non-faulty process and a "dead" faulty process. The analysis of Approximate Agreement herein is restricted to synchronous systems. These systems are representative of real time functions such as data sensor management and fault-tolerant clock synchronization [9, 13, 14, 18, 21].

2 Motivation

Recent research has addressed convergent voting in the simultaneous presence of multiple fault modes [2, 11, 10]. This work used the hybrid fault model of Thambidurai and Park, 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 [10, 11].

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 [8, 11, 13, 15, 20] *can not* 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, in a sense converting an omissive error into a more severe fault mode such as symmetric or asymmetric. However by the same 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. If the cause of receiving erroneous values is the link, it is assumed that the source process is faulty. 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. when 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 omission 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 [4, 5, 6, 17]. In addition, by a modest amount of internal self-checking or using *authenticated* messages [4, 24], the probability that a fault will behave omissively can be dramatically increased. Thus a corrupted message can be ignored, which has the same effect as if an omission error were received.

The reason that existing voting algorithms can not take advantage of omissive faults is due to the assumption that 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.
2. Locally diagnosed benign errors can not be discarded. Such errors are either used as received or are replaced with a default value.
3. The voting algorithms are less fault-tolerant.

Azadmanesh and Kieckhafer in [1, 3] showed that, if omissive faults remain omissive, the fault-tolerance is significantly improved. The results were derived for synchronous and asynchronous systems with complete connectivity. It was shown that the need to globally diagnose benign errors is decreased, thus reducing the overhead of running a voting algorithms to recognize global benign faults. Two subclasses of omissive faults were considered: *strictly omissive asymmetric* and *omissive symmetric*. A process which behaves in a strictly omissive asymmetric manner sends a single "correct" value to some processes and no value to others. Whereas, in omissive symmetric, the process value is received by no processes.

The research herein will employ all these failure modes, i.e. 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 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 to improve fault-tolerance rather than converting them into more severe failure modes.
2. There are no general methods for synchronous systems to measure the performance of different voting algorithms in the presence of omissive errors.
3. Omissive errors can be a predominant mode of failure in partially connected networks, for instance due to link failure, lost or delayed messages.
4. A partially connected system appears like a completely connected system with appropriate links behaving in omissive manner. Hence, understanding the impact of omissive faults will shed some light on the conditions necessary for global convergence in systems with partial connectivity.

Section 3 presents some background material and the traditional approaches to achieve convergence. It will also introduce the approach and the types of networks used herein to achieve convergence. Section 4 introduces omissions faults, and the impact of omission faults in the analysis of convergence rate. It shows how a partially connected system can logically look like a completely connected network. The section also develops the notation used in the rest of the paper, and using those notation to achieve convergence for two sub-families of voting algorithms called dynamic-σ and fixed-σ . Section 5 defines the two sub-families of algorithms. Sections 6 and 7 show the convergence rate and fault tolerance for the two sub-families of algorithms. Section 8 provides an example to better understand the process of determining whether convergence is possible, using the expressions obtained in the previous sections. Finally Section 9 summarizes the results and comments on future research prospects.

3 Partially Connected Systems

The vast majority of research in convergent voting has considered only completely connected systems [7, 8, 11, 13, 22, 23]. Thus, each process receives messages from all processes. 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 completely connected systems, local convergence, i.e. convergence between any two processes, implies convergence at the global level. However, in partially connected systems without the benefit of message relays, local convergence does not necessarily achieve global convergence. To achieve global convergence while reducing the overhead of message relays, two approaches have been identified. One approach employs special purpose communication hardware to increase the efficiency of handling relayed messages [16]. However, this approach does not reduce the order of complexity of message traffic. The other approach considers the system to be hierarchically composed of processor clusters [19]. Within each cluster, all processors are completely connected. One processor in each cluster is also connected to one processor in another cluster such that the set of clusters is completely connected. It is then possible to set one tolerance on agreement within a cluster, and a looser tolerance on agreement between clusters.

This paper presents a different approach to reducing the overhead of Approximate Agreement. It is assumed that the system in question is a large, regular, sparsely connected network of N processing nodes, each with degree d. It is assumed that the network prohibits the relay of messages. Thus, 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 number of messages passed by each link is constant. Thus, the elimination of message relays makes message passing overhead independent of number of nodes in the network, achieving significant saving for large systems.

While local convergence is a prerequisite to global convergence, it does not guarantee single step global convergence. For instance, two immediate neighbors such as processes i and j may receive values from their respective neighbors that are not shared by both processes i and j. Since these values change with every round, in the next round of voting, 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 [10]. This paper is mainly concerned with the conditions necessary to achieve local convergence, leaving global convergence for future work.

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. $\mathbf{P}_j$ is defined similarly for j.

$\mathbf{P}_{i\cap j}$ = $\mathbf{P}_i \cap \mathbf{P}_j$, the set of processes adjacent to both 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. Completely connected systems have the property that $\mathbf{P}_i = \mathbf{P}_{i\cup j} = \mathbf{P}_{i\cap j}$.

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

$\mathbf{U}_i$ = The submultiset of *correct* values received by process i, i.e. those values received by i from non-faulty processes.

$\mathbf{U}_{i\cup j}$ = The multiset of *correct* values generated in $\mathbf{P}_{i\cup j}$. In other words, $\mathbf{U}_{i\cup j}$ 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 $\mathbf{P}_j \backslash \mathbf{P}_i$.

χ = $|\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}| = |\mathbf{P}_j \backslash \mathbf{P}_{i\cap j}|$, the number of nodes adjacent to process i or j but not to both. In 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, $f_i \leq f$, where f_i is the number of erroneous values in $\mathbf{P}_i \backslash \mathbf{P}_{i\cap j}$ received by i.

As an example, Figure 1 shows the link connections for two processes i and j in an octagonal mesh. In this figure, χ 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 shown in the figure, while $\mathbf{P}_{i\cap j}$ consists of only those nodes within the dotted box.

In contrast to completely connected system, where $\mathbf{U}_{i\cap j} = \mathbf{U}_{i\cup j}$, the equality of these multisets are no longer preserved in partially connected system. In a partially connected system $\mathbf{U}_{i\cap j} \subset \mathbf{U}_{i\cup j}$. 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 Figure 1, intersection convergence between the processes i and j is obtained with respect to the values held by the processes within the dotted box, i.e. values held by processes $\mathbf{P}_{i\cap j}$. On the other hand, union convergence includes all the values for processes shown in the figure, i.e. the values used in the intersection convergence plus those values held by processes in the top and the bottom rows. It will be shown that union convergence is less restrictive than intersection convergence be-

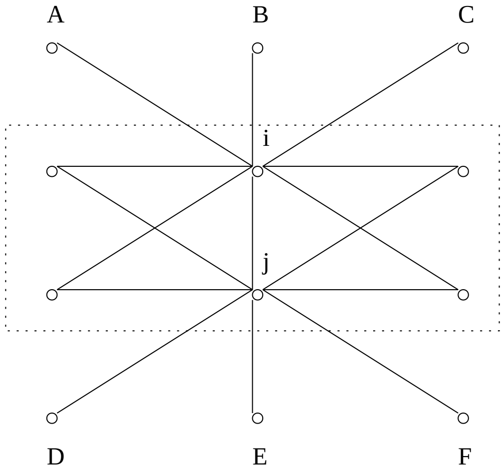

Figure 1: An Octagonal Mesh showing the link connections for nodes i and j.

cause union convergence will require less connectivity among processes. Also, the affects of union convergence diffuses across the overlapping regions of local convergence faster than that of intersection convergence simply because $|\mathbf{U}_{i \cup j}| \geq |\mathbf{U}_{i \cap j}|$. More specifically, because $|\mathbf{U}_{i \cup j}| = |\mathbf{U}_{i \cap j}| + (\chi - f)$, union convergence covers a larger subgraph than intersection convergence. To this end, we define two types of local convergence for partially connected systems:

Union Convergence: 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:

[U1] $F(\mathbf{V}_i) \in \rho(\mathbf{U}_{i \cup j})$, and $F(\mathbf{V}_j) \in \rho(\mathbf{U}_{i \cup j})$,

[U2] $|F(\mathbf{V}_i) - F(\mathbf{V}_j)| \leq C\delta(\mathbf{U}_{i \cup j})$, where $0 \leq C < 1$.

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

[I1] $F(\mathbf{V}_i) \in \rho(\mathbf{U}_{i \cap j})$, and $F(\mathbf{V}_j) \in \rho(\mathbf{U}_{i \cap j})$,

[I2] $|F(\mathbf{V}_i) - F(\mathbf{V}_j)| \leq C\delta(\mathbf{U}_{i \cap j})$, where $0 \leq C < 1$.

where:

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

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

$\rho(\mathbf{U}_{i\cap j})$ = $[\min(\mathbf{U}_{i\cap j}), \max(\mathbf{U}_{i\cap j})]$; the real interval spanned by $\mathbf{U}_{i\cap j}$. $\rho(\mathbf{U}_{i\cap j})$ is called the *range* of $\mathbf{U}_{i\cap j}$. $\rho(\mathbf{U}_{i\cup j})$ is defined similarly.

$\delta(\mathbf{U}_{i\cap j})$ = $\max(\mathbf{U}_{i\cap j}) - \min(\mathbf{U}_{i\cap j})$; the difference between the maximum and minimum values of $\mathbf{U}_{i\cap j}$. $\delta(\mathbf{U}_{i\cap j})$ is called the *diameter* of $\mathbf{U}_{i\cap j}$. $\delta(\mathbf{U}_{i\cup j})$ is defined similarly.

Another major difference between completely connected and partially connected systems is their handling of benign faults. Herein we distinguish between *local* and *global* benign faults. In a global benign error, the 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 [11]. 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 [15]. Therefore, in partially connected systems, only symmetric and asymmetric faults are considered.

4 Convergence with Omission Faults

In all previous studies of synchronous Approximate Agreement, it was required that the size of the voting multiset received be identical for all processes, i.e. $|\mathbf{V}_i| = |\mathbf{V}_j|\ \forall\ i, j \in \{1, \ldots, N\}$. Thus, if an omissive error occurred, it was required to substitute a default value for the missing value. This action had the effect of *transforming omissive errors into transmissive errors*. The only exception allowed was for a value transmitted by a globally recognized faulty process (i.e. a benign fault). This value could be discarded because it was assured *a priori* that all other non-faulty processes would do likewise, preserving the equality $|\mathbf{V}_i| = |\mathbf{V}_j|$. As a result, the previous voting algorithms could not exploit the omissive behavior of malicious faults. Similarly, a process could not simply disregard its locally diagnosed transmissive errors, such as parity errors. The process either had to use the erroneous value as received, or substitute the value with a default. This is again to ensure $|\mathbf{V}_i| = |\mathbf{V}_j|$.

Azadmanesh and Kieckhafer [1, 3] showed that by further partitioning symmetric and asymmetric faults into transmissive and omissive faults, MSR voting algorithms can be significantly more robust than voting algorithms under the previous models. What follows is a continuation of this work, showing that voting algorithms in synchronous partially connected system, under the same new fault partitioning [1, 3], will be more fault tolerant than that predicted under the previous fault models.

A variant of the MSR family [11] 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 not fixed and may change in every round of voting. Specifically, during a voting round, when an OMSR algorithm receives either an omissive error or a self-evident transmissive error, it simply discards that error. No defaults are substituted into the voting multiset $\mathbf{V}$. Thus, omissive errors remain omissive, and self-evident errors become omissive. As a result, the equality of $|\mathbf{V}_i|$ and $|\mathbf{V}_j|$ is no longer preserved.

4.1 Conversion to Complete Connectivity

The voting algorithms for partially connected systems use the same approximation function as that used for completely connected systems [1, 8, 11]:

$$F(\mathbf{V}_i) = \mathrm{mean}\left[Sel_{\sigma_i}(Red^{\tau}(\mathbf{V}_i))\right]. \tag{4.1}$$

The "Reduction" function Red^{τ} removes the τ largest and τ smallest elements from multiset $\mathbf{V}_i$, i.e. $Red^{\tau}(\mathbf{V}_i) = \langle v_{i,(1+\tau)}, \ldots, v_{i,(V_i-\tau)} \rangle$. The new submultiset is called *medial multiset* $\mathbf{M}_i$. The "Selection" function Sel_{σ_i} then selects a submultiset $\mathbf{S}_i$ of σ_i elements from the reduced multiset $Red^{\tau}(\mathbf{V}_i)$, i.e. $\mathbf{S}_i = Sel_{\sigma_i}(Red^{\tau}(\mathbf{V}_i)) = \langle s_{i,1}, \ldots, v_{i,\sigma_i} \rangle$. The final voted value is the arithmetic mean of the selected multiset.

With respect to two arbitrary processes such as i and j, a partially connected system can be thought of as a completely connected system. As an example, Figure 2 shows how a partially connected network such as Figure 1 is converted to look like a completely connected network.

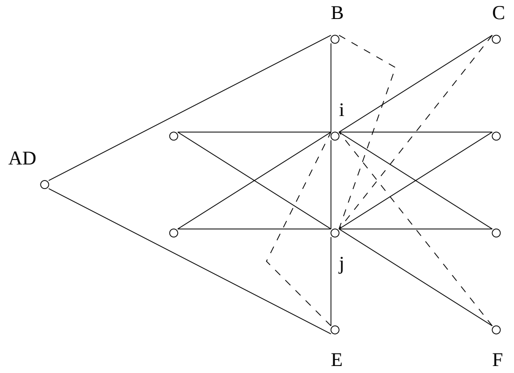

Figure 2: Completely connected view of Figure 1 with respect to nodes i and j

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 transmitted on these lines are lost. In essence, these lines are the source for omission faults. In this figure, 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 thought of a single faulty node sending conflicting data to i and j. This node is shown as AD in the figure. With respect to nodes B and C, they are behaving in strictly omissive asymmetric manner. Their correct values are sent to i but are not received by j. A similar situation exists for nodes E and F.

Under the same fault model, Azadmanesh and Kieckhafer [1] have already obtained the expressions for convergence rate and fault tolerance for completely connected systems. Thus, by being able to logically convert partial connectivity to complete connectivity, their results can be applied to partially connected networks. In [1], the parameters used to represent different fault modes are:

ω_{a1_i} = The number of strictly omissive asymmetric values in the network received by process i but not received by process j. The maximum number of such faults is ω_{a1}.

a'_i = The number of transmissive asymmetric faults in the network received by process i, i.e.

the number of faults displaying any form of asymmetric behavior other than strictly omissive. a' is the maximum number of such faults.

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

ω_s = The number of omissive symmetric faults in the network.

b = The number of benign faults in the network.

So, the total number of faults $t_{complete}$ in the network is:

$$t_{complete} = (a' + \omega_{a1}) + (s' + \omega_s) + b. \tag{4.2}$$

The relationships between this new and the previous fault partitioning which can not exploit omission faults is specified by the relations: $a = a' + \omega_{a1}$ 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, we see that:

$$\omega_{a1} = \omega_{a_{i\cap j}} + 2(\chi - f) \tag{4.3}$$

$$a' = a_{i\cap j} + f \tag{4.4}$$

$$s' = s_{i\cap j} \tag{4.5}$$

$$\omega_s = \omega_{s_{i\cap j}} \tag{4.6}$$

where:

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

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

By replacing the parameters in (4.2) with the expressions in (4.3) - (4.6), the total number of faults $t_{partial}$ in the network is:

$$t_{partial} = (a_{i\cap j} + \omega_{a_{i\cap j}}) + (s_{i\cap j} + \omega_{s_{i\cap j}}) + 2\chi - f \tag{4.7}$$

In a completely connected network, the benign faults can be discarded a priori before the voting algorithm executes, because the benign faults are globally diagnosed and thus every process is aware of them. Furthermore, a process does not receive values from processes which behave in symmetric omissive manner, ie. ω_s. Thus, the total number of values $N_{i,complete}$ received by a process i in a completely connected network is:

$$N_{i,complete} = (a'_i + \omega_{a1_i}) + s'. \tag{4.8}$$

Mapping this equation to Figure (2), we see that:

$$a'_i = a_{i,i\cap j} + f_i \tag{4.9}$$

$$\omega_{a1_i} = \omega_{a_{i,i\cap j}} + (\chi - f_i) \tag{4.10}$$

$$s' = s_{i\cap j} \tag{4.11}$$

where:

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

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

Note that the number of transmissive symmetric values $s_{i\cap j}$ in $\mathbf{P}_{i\cap j}$ must be the same as the number of transmissive symmetric values received by each process i and j. For this reason, no subscript is used to associate $s_{i\cap j}$ to either i or j. For the same reason $\omega_{s_{i\cap j}}$ is not associated to either of the processes.

Applying the expressions in (4.9) - (4.11) to (4.8), shows that the number of values received by process i is:

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

The following theorems take advantage of this conversion to a completely connected form to find the expressions for convergence rate and fault tolerance for the two sub-families of voting algorithms, i.e. dynamic-σ and fixed-σ , defined later.

4.2 Definition of γ

It will been shown that the convergence rate of an OMSR algorithm depends on a parameter called γ, which is a measure of how uniformly the elements of $\mathbf{S}$ are distributed within the medial multiset $\mathbf{M}$ [11].

By definition, the selected multiset $\mathbf{S}_i = \langle s_{i,1}, \ldots, s_{i,\sigma} \rangle$ is a *subsequence* of $\mathbf{V}_i = \langle v_{i,1}, \ldots, v_{i,V_i} \rangle$. Thus, each element of $\mathbf{S}_i$ corresponds to one unique element of $\mathbf{V}_i$. Let g be the index of any element of $\mathbf{S}_i$, and let $k_i(g)$ be the index of the corresponding element in $\mathbf{V}_i$. Then, for each $g \in \{1, \ldots, \sigma_i\}$ there exists exactly one $k_i(g) \in \{1, \ldots, V_i\}$ which guarantees that $s_{i,g} = v_{i,k_i(g)}$ *for all possible* $\mathbf{V}_i$.

Given two indices into $\mathbf{S}_i$, g and $h \in \{1, \ldots, \sigma_i\}$, 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 $(s_{i,g}, \ldots, s_{i,h})$ in $\mathbf{S}_i$. Thus, $\Delta k_i(g,h)$ is the number of elements of $\mathbf{M}_i$ in the submultiset $\langle m_{i,k_i(g)+1}, \ldots, m_{i,k_i(h)} \rangle$.

Finally, define $\gamma_{i,z}$ as the *minimum* value which ensures that $\Delta k_i(g, g+\gamma_{i,z}) \geq z$, for all $g \in \{1, \ldots, \sigma_i - \gamma_{i,z}\}$. By this definition, $\gamma_{i,z}$ exists only if $|\mathbf{M}_i| > z$. Furthermore, by the same definition, if $\gamma_{i,z}$ exists then $\gamma_{i,z} \leq \sigma_i$. The following pseudocode shows how to obtain $\gamma_{i,z}$:

```
γ_{i,z} = 0
DO
   ExitFor = FALSE
   FOR g = 1 TO σ_i − γ_{i,z} AND (NOT ExitFor)
      IF Δk_i(g, g + γ_{i,z}) < z THEN
         ExitFor = TRUE
   END FOR
   γ_{i,z} = γ_{i,z} + 1
UNTIL (γ_{i,z} = σ_i OR NOT ExitFor)
IF NOT ExitFor THEN
   γ_{i,z} = γ_{i,z} − 1
```

ELSE

PRINT("$\gamma_{i,z}$ does not exist")

4.3 Impacts of Fault Modes

For convenience, hereafter "strictly omissive asymmetric" faults are referred by "strictly omissive" faults.

In previous studies, an "error" is defined as any value received from a faulty process. However, even though a strictly omissive process is faulty, its values are "correct". We therefore redefine an "error" to be any value generated by a faulty process that does not behave in strictly omissive manner. 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 $F(\mathbf{V}_i)$ generates a value within the range of non-erroneous values.

It is possible that $a'_i \neq a'_j$, because OMSR voting algorithms no longer require processes to use default values for values not received. In addition, since two non-faulty processes may receive erroneous values from the same faulty process, it is conceivable that $(a'_i + a'_j) \geq a'$.

Due to the definition, ω_{a1_i} is the number of strictly omissive processes whose (correct) values are received by process i but not by process j. This definition implies that $\omega_{a1_i} \leq \omega_{a1}$. Similarly, $\omega_{a1_j} \leq \omega_{a1}$. In addition, if the value of every strictly omissive process is received by either i or j but not by both, then $\omega_{a1_i} + \omega_{a1_j} = \omega_{a1}$. Whereas, if the value of a strictly omissive process is received by neither process i nor j, but received by a third process, then $\omega_{a1_i} + \omega_{a1_j} < \omega_{a1}$. As a result, $(\omega_{a1_i} + \omega_{a1_j}) \leq \omega_{a1}$.

Recall that in Subsection 3, $\mathbf{U}_{i\cap j}$ is defined as the multiset of correct values generated within $\mathbf{P}_{i\cap j}$, i.e. the correct values generated by either process i or j in $\mathbf{P}_{i\cap j}$. Thus, it is important to observe that $|\mathbf{U}_{i\cap j}| \neq |\mathbf{U}_i \cap \mathbf{U}_j|$, but rather $|\mathbf{U}_{i\cap j}| = |\mathbf{U}_i \cap \mathbf{U}_j| + \omega_{a_{i,i\cap j}} + \omega_{a_{j,i\cap j}}$. This is in contrast to MSR algorithms, where $|\mathbf{U}_{i\cap j}| = |\mathbf{U}_i \cap \mathbf{U}_j|$, regardless of whether the system is completely or partially connected.

Since, the work herein is focused on local convergence, in (4.7), $t_{partial}$ is the total number of faults with respect to two adjacent non-faulty processes such as i and j, and not the total number of faults in the system. However, for global convergence, two alternative can be identified. One is to place a limit on the number of faults in the entire system, as done in [16]. 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. Global convergence under both approaches are under investigation.

5 Fixed and Dynamic Voting Algorithms

It is the selection function that determines the values of σ_i and σ_j. As indicated, in previous studies of voting algorithms, it is presumed that all voting multisets must carry the same number of elements in every round of voting, which implies that $|\mathbf{M}_i| = |\mathbf{M}_j|$. Hence, regardless of the selection function used, the number of elements selected is always *fixed* and is known a priori, i.e. $|\mathbf{S}_i| = |\mathbf{S}_j|$. In addition, for any k^{th} elements $s_{i,k}$ and $s_{j,k}$, the positions of their corresponding elements into $\mathbf{M}_i$ and $\mathbf{M}_j$ are the same, i.e. if $s_{i,k} = m_{i,\ell}$ and $s_{j,k} = m_{j,p}$, then $\ell = p$. As a result, *any* selection function will perform *uniformly* on $\mathbf{M}_i$ and $\mathbf{M}_j$. Because of these reasons, there was no need to distinguish among voting algorithms.

In OMSR, however, the size $\mathbf{V}$ for different processes need not be of the same fixed size. This implies that their medial multisets might be of different sizes. Thus, depending on the selection function, the number of selected elements might be different from one process to another. This and the fact that not all selection functions belong to the same family of algorithms, caused the need to distinguish among different families of algorithms. There are two general families of selection functions: *Fixed-σ* and *Dynamic-σ*. A Fixed-σ selection function always selects the same number of entries from the multiset $\mathbf{M}$, regardless of the size of $\mathbf{M}$. For instance, Fault-Tolerant Midpoint [7] selects exactly two values, i.e. the two extreme values, m_1 and m_M from $\mathbf{M}$. By contrast, in a Dynamic-σ selection function, the number of entries selected from $\mathbf{M}$ depends on the size of $\mathbf{M}$, i.e. for process i, σ_i is a function of M_i. There are several known Dynamic-σ selection functions, for example, the Fault-Tolerant Mean [7], Dolev's Optimal algorithm [8], and the Mixed-Mode Optimal algorithm [11].

Each family of algorithms could contain many sub-families. Two sub-families are considered for OMSR, one from each family, in such a way to ensure that together these sub-families encompass all commonly used voting algorithms.

5.1 Dynamic-σ Selection Functions

There could be many sub-families of Dynamic-σ selection functions. This study is limited to the sub-family of *enumerative* selection functions defined below. In an enumerative selection function, the relative position of each *selected* element in $\mathbf{M}$ *does not* depend on M. More formally, for two different size medial multisets $\mathbf{M}_i$ and $\mathbf{M}_j$, if $s_{i,g} = m_{i,k}$, then $s_{j,g} = m_{j,k}$, $\forall\ k \in \{1, \ldots, \min(M_i, M_j)\}$.

Let us define the medial multiset $\mathbf{M}_{max} = \langle m_1, \ldots, m_{M_{max}} \rangle$, where M_{max} is the maximum possible size of all medial multisets in the system, and $\mathbf{S}_{max} = Sel_{\sigma_{max}}(\mathbf{M}_{max})$, where σ_{max} is the number of elements selected from $\mathbf{M}_{max}$. Also, let us define the enumerative selection set as a set of integers $E = \{e_1, \ldots, e_{\sigma_{max}}\}$, where $e_j < e_{j+1}\ \forall\ j \in \{1, \ldots, \sigma_{max} - 1\}$. Set E is then a list of the indices of all elements of $\mathbf{M}_{max}$ selected for inclusion in $\mathbf{S}_{max}$. Thus, $s_j = m_{e_j}$.

By this definition, for any process i, the selected multiset $\mathbf{S}_i$ is then: $Sel_{\sigma_i}(\mathbf{M}_i) = \langle m_{e_1}, \ldots, m_{e_{\sigma_i}} \rangle$, where σ_i is the largest value such that $e_{\sigma_i} \leq |\mathbf{M}_i|$, i.e. $\sigma_i = \max(j : e_j \leq M_i)$. In other words, the elements selected from $\mathbf{M}_i$ are those whose indices appear in E.

5.2 Fixed-σ Selection Functions

Recall that a selection function is a mapping from the set $\{1, \ldots, \sigma\}$ to the codomain set $\{k(1), k(2), \ldots, k(\sigma)\}$. In a Fixed-$\sigma$ election function, $\sigma_i = \sigma_j$, regardless of the sizes of $\mathbf{M}_i$ and $\mathbf{M}_j$. This will make the convergence rate expression simpler because $\sigma_{max} = \sigma_{min}$. However, for two arbitrary processes i and j, if $M_i \neq M_j$, the sets of codomain values obtained for i and j may not be the same. Specifically, $k_i(g) \neq k_j(g), \ \forall\, g \in \{1, \ldots, \sigma\}$. This problem does not exist in dynamic-σ selection functions. The subfamily of fixed-σ adopted has the following properties:

$$M_i \geq M_j \Rightarrow \begin{cases} k_i(g) \geq k_j(g), & \forall\, g \in \{1, \ldots, \sigma\} \\ \\ \Delta k_i(g, g+1) \geq \Delta k_j(g, g+1), & \forall\, g \in \{1, \ldots, \sigma - 1\} \end{cases} \tag{5.1}$$

Informally, these properties state that as 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.

The remainder of this paper is devoted to analyzing the necessary conditions for Union-Convergence, Intersection-Convergence, and their fault-tolerance under the two sub-families of selection functions described. The results will show improvement in fault tolerance in comparison to that of Thambidurai and Park [20] and MSR fault-models.

6 Dynamic-σ Convergence Rate

6.1 Union Convergence

THEOREM 1 : *Assume a regular, partially connected system. Given the voting algorithm $F(\mathbf{V})$, an enumerative dynamic-σ selection function, and two multisets $\mathbf{V}_i$ and $\mathbf{V}_j$, such that $F(\mathbf{V}_i) \geq F(\mathbf{V}_j)$, the convergence rate is:*

$$C \leq \begin{cases} \frac{\gamma_{j,a_{j,i\cap j}}+\omega_{a_{j,i\cap j}}+\chi}{\sigma_j} & : \ \sigma_i \leq \sigma_j \\ \frac{\sigma_i-\sigma_j+\gamma_{j,a_{j,i\cap j}}+\omega_{a_{j,i\cap j}}+\chi}{\sigma_i} & : \ \sigma_i \geq \sigma_j \end{cases} \quad (1)$$

PROOF: It has been shown [1] that in a completely connected network, the convergence rate is:

$$C \leq \begin{cases} \frac{\gamma_{j,a'_j}+\omega_{a1_j}}{\sigma_j} & : \ \sigma_i \leq \sigma_j \\ \frac{\sigma_i-\sigma_j+\gamma_{j,a'_j}+\omega_{a1_j}}{\sigma_i} & : \ \sigma_i \geq \sigma_j \end{cases}$$

Since we indicated that a partially connected system, from 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 omission fault, we can safely use (4.9) and (4.10). Thus, by replacing a'_j with $a_{j,i\cap j}+f_j$, and ω_{a1_j} with $\omega_{a_{j,i\cap j}}+(\chi-f_j)$, the convergence rate in 1 is obtained. □

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

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

PROOF: For a completely connected system, it has been shown that [1]:

$$V_j \ \geq \ 2\tau+\max(a'_j+\omega_{a1_j}+1,\sigma_j), \quad \tau \geq a'+s' \quad (2)$$

It was indicated in (4.9) - (4.10) and (4.4) - (4.5) that:

$$\begin{aligned} a'_j &= a_{j,i\cap j}+f_j \\ \omega_{a1_j} &= \omega_{a_{j,i\cap j}}+(\chi-f_j) \\ a' &= a_{i\cap j}+f \\ s' &= s_{i\cap j} \end{aligned}$$

Applying these expressions to (2), we obtain:

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

□

THEOREM 3 : *Given a partially connected network as described, and an enumerative dynamic-σ selection, a Union convergent voting algorithm exists if the following is true:*

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

PROOF: Let the number of nodes in Figure (1) be $N_{partial}$. When this figure is converted to the completely connected form, i.e. Figure (2), the number of nodes is $N_{partial} - f$. It has been shown [1] that a completely connected network with $N_{complete}$ nodes must satisfy the following expression for the algorithm to be convergent:

$$N_{complete} \geq 3a' + 2s' + \omega_{a1} + \omega_s + b + 1 \tag{2}$$

Using this inequality against equations (4.3) - (4.6), 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:

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

Now, since $\mathbf{P}_{i\cap j} = N_{partial} - 2\chi$, (3) becomes:

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

□

MSR algorithms in partially connected systems do not distinguish between transmissive and omissive faults. As a result, all strictly omissive 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 [10] have shown that a convergent voting algorithm exists if:

$$\mathbf{P}_{i\cap j} \geq 3a_{a_{i\cap j}+\omega_{a_{i\cap j}}} + 2s_{s_{i\cap j}+\omega_{s_{i\cap j}}} + 2f + 1 \tag{6.1}$$

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

$$\begin{aligned} \mathbf{P}_{i\cap j} &\geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f + 1 \\ &= 3(a_{i\cap j} + \omega_{a_{i\cap j}}) + 2(s_{i\cap j} + \omega_{s_{i\cap j}}) + 2f + 1 - (2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}) \\ &= 3a_{a_{i\cap j}+\omega_{a_{i\cap j}}} + 2s_{s_{i\cap j}+\omega_{s_{i\cap j}}} + 2f + 1 - (2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}}) \end{aligned} \tag{6.2}$$

Comparing (6.2) to (6.1), it is observed that including omission faults into the fault model reduces $\mathbf{P}_{i\cap j}$ by $(2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}})$. Since in large geographically networks, omissions can be the dominant mode of failure, the saving of $(2\omega_{a_{i\cap j}} + \omega_{s_{i\cap j}})$ 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 $\mathbf{V}$ and not use it in the execution phase. 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 the result, employing OMSR algorithms reduces the need to globally diagnose errors.

6.2 Intersection Convergence

Intersection-convergence requires a more restrictive criterion than union-convergence. Specifically, due to the definition of intersection-convergence, $F(\mathbf{V})$ 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 τ 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 $\rho(\mathbf{U}_{i\cap j})$. Accordingly, $\tau \geq a_{i\cap j} + s_{i\cap j} + \chi$.

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

It should be noted that, although the expressions for union and intersection convergence are the same, they may not produce the same results because τ for the two environments are different. That will affect the medial multisets and in turn different σ values will be produced for each environment.

7 Fixed-σ Selection Functions

Recall that in fixed-σ, the number of elements selected is fixed, regardless of the size of $\mathbf{V}$. Therefore, the expression for convergence rate is simpler. Also, the method for obtaining the convergence rate is the same as that of dynamic-σ, with the difficulty that no longer $k_i(\ell) = k_j(\ell)$. The next two theorems show the convergence rate and fault-tolerance expressions for union convergence. As in the dynamic-σ case, the results for the intersection convergence are similar to the union convergence by replacing f with χ.

THEOREM 4 : *Assume a regular, partially connected system. Given the voting algorithm $F(\mathbf{V})$, and a fixed-σ selection function, the Union convergence rate is:*

$$C = \frac{\gamma_{a_{i \cap j} + \omega_{a_{i \cap j}} + \chi}}{\sigma} \tag{1}$$

PROOF: For a completely connected system, [1] has shown that the convergence rate is:

$$C = \frac{\gamma_\alpha}{\sigma} \tag{2}$$

where α represents the maximum number of effective number of asymmetric values seen by any non-faulty process. Consider two processes i and j as in Figure (1). The maximum effective number of asymmetric values seen by a process 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 received from the periphery processes, i.e. those processes whose values are not received by process j, is χ. Thus, the total number of effective asymmetric values is $(a_{i\cap j}+\omega_{a_{i\cap j}}+\chi)$. By replacing this expression for α in (2), (1) is obtained. □

THEOREM 5 : *Given a partially connected network as described, and a fixed-σ selection function, a Union convergent voting algorithm exists if the following is true:*

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

PROOF: The proof is similar to Theorem 3. □

8 Application Example

This example uses different values for the fault modes to show when Union convergence is possible. A portion of a regular network is shown in Figure (3). In this network, four nodes are labeled i, j, k, and l, showing the connections 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 $\mathbf{P}_i\backslash\mathbf{P}_{i\cap j}$, and use $f_{i,j}$ to show the faulty processes in $\mathbf{P}_i\backslash\mathbf{P}_{i\cap j}$. By inspecting the figure, we can see that:

$$\begin{aligned}\mathbf{P}_{i\cap j} &= 12\\ \mathbf{P}_{j\cap k} &= 10\\ \chi_{i,j} &= 7\\ \chi_{j,k} &= 9\end{aligned}$$

On the other hand, according to Theorem 3, i and j are convergent if:

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

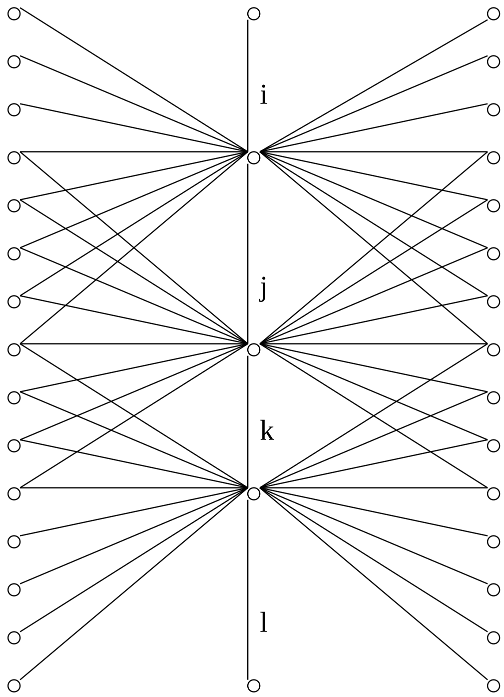

Figure 3: A partially connected network

and j and k are convergent if:

$$\mathbf{P}_{j\cap k} \geq 3a_{j\cap k} + 2s_{j\cap k} + \omega_{a_{j\cap k}} + \omega_{s_{j\cap k}} + 2f_{j,k} + 1$$

Thus, convergence between i and j, and between j and k is possible if:

$$12 \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + \omega_{s_{i\cap j}} + 2f_{i,j} + 1 \quad (8.1)$$

$$10 \geq 3a_{j\cap k} + 2s_{j\cap k} + \omega_{a_{j\cap k}} + \omega_{s_{j\cap k}} + 2f_{j,k} + 1 \quad (8.2)$$

Using the conditions in (8.1) and (8.2), Table (1) uses some examples showing when convergence is possible between i and j, and between j and k.

This example shows that, while a pair of nodes are convergent with respect to each other's correct values, a different pair of nodes may not be convergent. For instance, in row 4 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 peculiarity, however, is not necessarily an indication that the entire network is divergent. As long as $\tau \geq a' + s' + f$, the voted values of each

Row #	$a_{i\cap j}$ $a_{j\cap k}$	$s_{i\cap j}$ $s_{j\cap k}$	$\omega_{a_{i\cap j}}$ $\omega_{a_{j\cap k}}$	$\omega_{s_{i\cap j}}$ $\omega_{s_{j\cap k}}$	$f_{i,j}$ $f_{j,k}$	Convergence ? Inequality (8.1)	Convergence ? Inequality (8.2)
1	2	1	0	1	1	Yes	No
2	2	2	0	0	0	Yes	No
3	1	2	0	2	1	Yes	No
4	0	3	0	3	1	Yes	No
5	0	0	0	9	0	Yes	Yes
6	0	0	4	4	0	Yes	Yes
7	3	0	0	0	0	Yes	Yes
8	3	0	1	1	0	Yes	No
9	3	1	0	0	1	No	No
10	2	3	0	0	0	No	No

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

node will be within the range of the correct values in the entire network. As a result, nodes maybe 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 network.

Row 7 shows that 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 faults in $\mathbf{P}_{k\cap \ell}$. For this distribution of faults, all pairs, i.e. (i, j), (j, k), and (k, ℓ) are convergent, and each process will generate values within the range of the correct values in the entire network. This implies that there must be at least 9 such faults in the network. However, if any process encounters that many faults, there is no guarantee that the process will produce values within the range of the correct values in the entire network. This is because $\tau = 9$ implies that $\mathbf{M} = \mathbf{V} - 2\tau = 19 - 18 = 1$. As a result γ does not exist. Specifically, by definition, for γ to possibly exist, $|\mathbf{M}|$ must be at least two. Once one process becomes divergent in the network, it may never generate values within the correct values in the network again. Thus, that process would act exactly like a faulty process, even though it is not faulty. As a result, the process may infect other processes which in turn may infect other processes. As a result, the entire network may never become convergent. This example points out that global convergence is very dependent upon the distribution of faults within the network. The only way to prevent this problem is to set a limit on the number of faults in the network while making sure that every process can stay convergent within this limit of faults. This is however restrictive because in a large network the limit will then be determined by the condensity of the connections between each pair of processes rather than the number of nodes in the entire network.

9 Summary and Conclusions

This paper has demonstrated the analytical power of hybrid fault models in the presence of omission faults. It was shown that the inclusion of omission faults improves fault tolerance in comparison to other models such as MSR.

Obtaining convergence directly from partial connectivity is a very tedious process because the voting multisets are not of equal sizes, which introduces a number of difficulties. This is in contrast to traditional analyses which presume equality in voting multisets among all processes. 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 in the network. This is already under investigation. If this conjecture is true, then the performance of global convergence can be obtained.

This paper considered two subfamilies of algorithms called dynamic-σ and fixed-σ . Since in presence of omission faults, 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-σ but not to dynamic-σ , or Fault-Tolerant Mean belongs to dynamic-σ but not to fixed-σ .

Table II summarizes the convergence bounds for fixed-σ and dynamic-σ 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.

Union	Intersection
$\tau \geq a' + s' + f$ $M_{max} = \left\|\mathbf{P}_{i\cap j}\right\| + \chi - 2\tau$ $P_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + 2f + 1$ $C \leq \left[\frac{\gamma_{j,a_{j,i\cap j}} + \omega_{a_{j,i\cap j}} + \chi}{\sigma_j} : \sigma_i \leq \sigma_j, \quad \frac{\sigma_i - \sigma_j + \gamma_{j,a_{j,i\cap j}} + \omega_{a_{j,i\cap j}} + \chi}{\sigma_i} : \sigma_i \geq \sigma_j\right]$	$\tau = a' + s' + \chi$ $M_{max} = \left\|\mathbf{P}_{i\cap j}\right\| + \chi - 2\tau$ $P_{i\cap j} \geq 3a_{i\cap j} + 2s_{i\cap j} + \omega_{a_{i\cap j}} + 2\chi + 1$ $C \leq \frac{\gamma_{a_{i\cap j}} + \omega_{a_{i\cap j}} + \chi}{\sigma}$

Table II: Summary of Necessary Dynamic-σ and Fixed-σ Convergence Parameters

References

[1] M.H. Azadmanesh, R.W. Kieckhafer, "Exploiting Omissive Faults in Synchronous Approximate Agreement" *ACM Transactions on Computer Systems*, submitted.

[2] M.H. Azadmanesh, R.W. Kieckhafer, *The General Convergence Problem: A Unification of Synchronous and Asynchronous Systems*, Dependable Computing and Fault-Tolerant Systems, Vol. 9, Springer-Verlag, pp. 251-267, 1995.

[3] M.H. Azadmanesh, R.W. Kieckhafer, *New Hybrid Fault Models for Asynchronous Approximate Agreement*, IEEE Trans. Computers, Vol. 45, No. 4, pp. 439-449, April 1996.

[4] Cristian, F., Aghili, H., Strong, R., and Dolev, D., Atomic Broadcast: From Simple Message Diffusion To Byzantine Agreement. *Proc. Fifteenth International Symposium on Fault-Tolerant Computing*, Jun. 1985, pp. 200-206.

[5] Cristian, F., Aghili, H., and Strong, R., Clock Synchronization in the Presence of Omission and Performance Faults, and Processor Joins. *Sixteenth International Conference on Fault-Tolerant Computing*, Vienna, Austria, 1986.

[6] Cristian, F., Dolev, D., Strong, R., and Aghili, H., Atomic Broadcast in a Real-Time Environment. *IBM Research Report, Almaden Research Center*, 1989.

[7] Dolev, D., Lynch, N.A., Pinter, S.S., Stark, E.W., and Weihl, W.E., Reaching Approximate Agreement in the Presence of Faults. *Proc. Third Symposium on Reliability in Distributed Software and Database Systems*, Oct. 1983.

[8] Dolev, D., Lynch, N.A., Pinter, S.S., Stark, E.W., and Weihl, W.E., Reaching Approximate Agreement in the Presence of Faults. *JACM* **33**, 3 (Jul. 1986), 499-516.

[9] Kieckhafer, R.M., Walter, C.J., Finn, A.M., and Thambidurai, P.M., The MAFT Architecture for Distributed Fault-Tolerance. *IEEE Trans. Comput.* **C-37**, 4 (Apr. 1988), 398-405.

[10] Kieckhafer, R.M., and Azadmanesh, M.H., Low Cost Approximate Agreement in Partially Connected Networks. *Journal of Computing and Information* **3**, 2 (1993), 53-85.

[11] Kieckhafer, R.M., and Azadmanesh, M.H., Reaching Approximate Agreement With Mixed Mode Faults. *IEEE Trans. on Parallel Distributed Systems* **5**, 1 (Jan. 1994), 53-63.

[12] Lamport, L., et al, The Byzantine Generals Problem. *ACM TOPLAS* **4**, 3 (Jul. 1982), 382-401.

[13] Lamport, L., Melliar-Smith, P.M., Synchronizing Clocks in the Presence of Faults. *JACM* **32**, 1 (Jan. 1985), 52-78.

[14] Lundelius, J., and Lynch, N., A New Fault-Tolerant Algorithm for Clock Synchronization. *Third Symposium on Principles of Distributed Computing*, Aug. 1984, pp. 75-87.

[15] Meyer, F.J., and Pradhan, D.K., Consensus with Dual Failure Modes. *Proc. Seventeenth Fault-tolerant Computing Symposium*, Jul. 1987, pp. 48-54.

[16] Ramanathan, P., et al, "Hardware-Assisted Software Clock Synchronization for Homogeneous Distributed Systems", *IEEE Trans. Comput.*, V. C-39, No. 4, pp. 514-524, Apr 1990.

[17] Schneider. F.B., "Byzantine Generals in Action: Implementing Fail-stop Processors", *ACM Trans Comp Sys*, Vol. 2, No. 2, pp. 145-154, May 1984.

[18] Schneider, F.B., *Understanding Protocols for Byzantine Clock Synchronization.* Report No. 87–859, Dept of Computer Science, Cornell University, Aug. 1987.

[19] Shin, K.G., and P. Ramanathan, "Clock Synchronization of a Large Multiprocessor System in the Presence of Malicious Faults", *IEEE Trans. Comput.*, V. C-36, No. 1, pp. 2-12, Jan 1987.

[20] Thambidurai, P.M., and Park, Y.K., Interactive Consistency with Multiple Failure Modes. *Proc. Seventh Reliable Distributed Systems Symposium*, Oct. 1988.

[21] Thambidurai, P.M., Finn, A.M., Kieckhafer, R.M., and Walter, C.J., Clock Synchronization in MAFT. *Proc. Nineteenth Fault-Tolerant Computing Symposium*, Jun. 1989, pp. 142-151.

[22] Vasanthavada, N., and Marinos, P.N., Synchronization of Fault-Tolerant Clocks in the Presence of Malicious Failures. *IEEE Trans. Comput.* **C-37**, 4 (Apr. 1988), 440-448.

[23] Vasanthavada, N., and Thambidurai, P., Design of Fault-Tolerant Clocks with Realistic Assumptions, *Proc. Eighteenth Fault-Tolerant Computing Symposium*, Jun. 1989, pp. 128-133.

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