ASYNCHRONOUS APPROXIMATE AGREEMENT IN PARTIALLY CONNECTED NETWORKS

M.H. Azadmanesh and R.M. Kieckhafer

Abstract – An important problem in fault-tolerant distributed systems is maintaining agreement between non-faulty processes in the presence of undiagnosed faults. To achieve agreement, processes must exchange their local “opinions” of a particular value, and then vote on the values received to arrive at a “consensus”. Approximate Agreement defines a condition in which it is not necessary for consensus values to be identical. Rather, it is only necessary that they agree to within a predefined tolerance.

Approximate Agreement can be achieved through a sequence of convergent voting rounds, in which the range of values held by non-faulty processes is reduced in each round. Simple expressions have been obtained for the convergence rate and fault tolerance of a broad family of “locally” convergent voting algorithms, in the presence of mixed fault-modes. However, the results apply only to synchronous systems, in which there is a known finite bound on computation and communications times.

This paper addresses convergent voting algorithms in asynchronous partially connected systems, in which no bound exists on process operations and communication delays. It is shown that these systems can be treated as a variant of synchronous partially connected systems. The result is tighter bounds on convergence rates and fault-tolerance than possible with previous analysis of asynchronous systems.

Key Words – Approximate Agreement, Byzantine Agreement, Clock Synchronization, Convergent Voting Algorithms, Fault-Tolerant Multiprocessors, Hybrid Fault Models.

1University of Nebraska at Omaha, Dept. of Computer Science, Omaha, Ne 68182-0500. E-mail: azad@alvaz.unomaha.edu
2Michigan Technological University, Dept. of Electrical and Computer Engineering, Houghton, MI 49931-1295. E-mail: rmkieckh@mtu.edu
1 INTRODUCTION

An important issue in fault-tolerant distributed computing is the ability of non-faulty processes to reach agreement on data values in the presence of faulty processes. This issue arises whenever non-faulty processes legitimately form differing “opinions” regarding the correct value. They must then exchange and vote upon their local values to arrive at a single consensus value. The problem is significantly more complex if a faulty process is permitted to send conflicting values to different non-faulty processes. A faulty process with this property has been called malicious, two-faced, Byzantine, or asymmetric.

In many real world applications, such as sensor data management and fault-tolerant clock synchronization [1, 2, 3, 4], non-faulty processes need not achieve exact agreement. Rather, they need only agree on a value to within a specified tolerance. This criterion is known as Approximate Agreement. Given an arbitrarily small positive real value \( \epsilon \), Approximate Agreement is defined by two conditions [5]:

A1: AGREEMENT – The voting algorithms executed by all non-faulty processes eventually halt with voted values that are within \( \epsilon \) 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.

Several algorithms for achieving Approximate Agreement have been published. Many employ multiple rounds of message exchange with convergent voting algorithms which guarantee that the range of values held by the non-faulty processes is reduced in each round [2, 5, 6, 7, 8, 9]. This property, called single-step convergence, guarantees that the range of values will eventually be less than \( \epsilon \), given enough rounds.

In the context of convergent voting, systems are partitioned into two classes: synchronous and asynchronous. In a synchronous system there are finite bounds on the processing and communication delays of non-faulty processes [5]. There is thus a point in time by which
any process executing a convergent voting algorithm will have received all data from all non-faulty processes. Any data arriving after that time must have come from a faulty process. By contrast, an asynchronous system imposes no finite bounds on process operation [5]. It is thus impossible to differentiate between a slow non-faulty process and a “dead” faulty process.

Recent research has addressed convergent voting in the simultaneous presence of multiple fault modes [6]. Faults were partitioned into three modes: asymmetric (Byzantine), symmetric (single-valued) and benign (self-incriminating). Using this mixed-mode 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. These results have been extended to analyze convergence in partially connected systems without the benefit of message relays [7]. However, this analysis was limited to synchronous systems.

This paper analyzes the convergence and fault tolerance properties of MSR voting algorithms in asynchronous partially connected systems. Section 2 presents some background material necessary to understand the convergent voting process. Section 3 discusses the partially connected networks, the definitions and assumptions used for the type of networks used in this paper. Section 4 reviews the results of [7] for synchronous partially connected systems, which are the basis for the rest of the paper. In Section 5, the techniques of [7] are extended to analyze convergent voting in asynchronous partially connected systems. Analysis of the two system types are tied together by showing that asynchronous systems can be treated as a variant of synchronous systems. This section will also provide an example showing the performance and fault tolerance for three common mesh networks. The last section will present a summary of the paper, and will give some discussion on the complexity of partially connected networks with regard to convergence. The paper concludes with some brief information about the recent research in partially connected networks.
2 BACKGROUND

2.1 Real-Valued Multisets

Approximate Agreement requires the manipulation of multisets of real values. A multiset is a collection of objects similar in concept to a set. However, it differs from a set in that all elements of a multiset are not necessarily distinct. A multiset of real numbers can be represented as a monotonically increasing sequence of the real values of its elements, i.e. $V = \langle v_1, \ldots, v_V \rangle$ is ordered such that: $v_i \leq v_{i+1} \forall i \in \{1, \ldots, V-1\}$ [10]. The size of $V$ is $V = |V|$. Among other operations, the following operations are used in this paper:

\[
\rho(V) = [\min(V), \max(V)] = [v_1, v_V]; \text{ the real interval spanned by } V. \rho(V) \text{ is called the range of } V.
\]

\[
\delta(V) = \max(V) - \min(V) = v_V - v_1; \text{ the difference between the maximum and minimum values of } V. \delta(V) \text{ is called the diameter of } V.
\]

\[
\text{mean}(V) = \text{The arithmetic mean of the real values of all elements of } V.
\]

**Union:** Let $W = V \cup U$. Then $W(r) = \max [V(r), U(r)] \forall r \in \mathbb{R}$.

**Intersection:** Let $W = V \cap U$. Then $W(r) = \min [V(r), U(r)] \forall r \in \mathbb{R}$.

**Subsequences:** Consider two non-empty multisets $U$ and $V$, where $U \subseteq V$. $U$ is a subsequence of $V$ if there is an order-preserving one-to-one mapping $k$, from the indices of $U$ to the indices of $V$, i.e. $u_j = v_{k(j)} \forall j \in \{1, \ldots, U\}$ and $k(j) < k(j + 1) \forall j \in \{1, \ldots, U - 1\}$.

2.2 Hybrid Fault Models

Early studies of distributed agreement algorithms assumed that all faults behave in a completely arbitrary or Byzantine manner [2, 5]. In real-world systems, however, truly Byzantine behavior occurs only under highly improbable conditions. Rather, the faults present are usually of mixed severity. Thus, the assumption that all faults are worst-case yields overly
conservative estimates of fault-tolerance and reliability. To account for this fact, hybrid fault models have been developed which allow faults of different severities to co-exist in the system, rather than treating all faults as Byzantine.

Accordingly, Meyer and Pradhan [11] have partitioned the space of all possible faults into two distinct modes: Benign faults, defined as those which are self-incriminating, or immediately self-evident to all non-faulty processes, and Malicious faults, defined as all faults which do not qualify as benign. Thambidurai and Park [12] have further partitioned malicious faults into two sub-modes: Symmetric faults, whose behavior is perceived identically by all non-faulty processes, and Asymmetric faults, whose behavior may be perceived differently by different non-faulty processes.

Given $a$ asymmetric, $s$ symmetric, and $b$ benign faults, the total number of faults is $t = a + s + b$. The three-mode fault model has been applied to both Byzantine Agreement [12, 13, 14, 15] and synchronous Approximate Agreement [6, 7, 16, 17]. By exploiting the relative rarity of asymmetric faults, this model permits more accurate analysis of the performance and fault-tolerance of agreement algorithms. Specifically, unless all faults are worst-case Byzantine faults, the voting algorithms are more fault-tolerant than predicted under the single-mode Byzantine fault model.

Plunkett and Fekete [18] also used a three-mode fault model consisting of asymmetric, symmetric, and omission faults. An omission occurs when a process does not receive a value that is expected. This model, although different in the approach of analyzing the Approximate Agreement problem, is a contraction of a five-mode fault model (named OTH-5), developed by Azadmanesh and Kiechlafer [19]. The OTH-5 fault model explicitly partitions both the asymmetric and symmetric fault modes of Thambidurai and Park into transmissive and omission sub-modes. A transmissive fault is one which delivers erroneous value(s) to one or more receiving processes. By contrast, an omission fault fails to deliver any value to one or more receiving processes, but does not deliver an erroneous value to any process.

This paper is concerned with the three-mode fault model of Thambidurai and Park, because convergence in partially connected networks under this fault model is still under investiga-
tion. Later studies may consider a more complex fault model such as OTH-5.

2.3 MSR Voting Algorithms

One large family of convergent voting algorithms has the general form [5, 6]:

\[ F(V) = \text{mean} \left[ Sel_{\sigma} (Red^\tau (V)) \right] . \]

The reduction function \( Red^\tau \) removes the \( \tau \) largest and \( \tau \) smallest elements from multiset \( V \). The function \( Sel_{\sigma} \) selects a submultiset of \( \sigma \) elements from the reduced multiset \( Red^\tau (V) \). The final voted value is the arithmetic mean of the selected multiset.

If \( Sel_{\sigma} \) produces a subsequence of \( Red^\tau (V) \), then \( F(V) \) is the Mean of a Subsequence of the Reduced multiset. The family of all voting algorithms with this property are called Mean-Subsequence-Reduced (MSR) algorithms [6]. Members of the MSR family differ from each other only in their definition of the selection function \( Sel_{\sigma} \). Some examples of MSR algorithms are [5, 6]: the Fault-Tolerant Midpoint, the Fault-Tolerant Mean, the Mixed-Mode Optimal algorithm, the Binary Mean, and the Binary Suboptimal algorithm.

Throughout this paper the following definitions are used frequently in the context of MSR algorithms. For simplicity of notation, subscripts may be omitted when there is no room for confusion.

\[ V_k = \text{The multiset of values received in a given round by non-faulty process } k. \]

\[ M_k = Red^\tau (V_k), \text{ the medial multiset of } V_k. \]

\[ S_k = Sel_{\sigma}(M_k), \text{ the Selected Multiset generated by } F(V_k). \text{ The number of selected elements } \sigma \text{ is identical for all non-faulty process.} \]
3 Partially Connected Systems

The vast majority of research in convergent voting has considered only completely connected systems [2, 5, 6, 8, 9]. Thus, any process can receive messages from all other 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.

Convergent voting has been studied in partially connected synchronous systems without the benefit of message relays [7]. In such systems a given process does not receive values from all non-faulty processes. Rather, it receives values from a specific subset of processes. There are thus two types of convergence to be considered: local convergence over a specified subgraph, and global convergence over the entire system graph. This paper is concerned with theoretical bounds on the ability to achieve local convergence. The problem of achieving global convergence given local convergence is an open question which is currently under investigation [7, 20, 21].

3.1 Assumptions

The following are assumed regarding partially connected systems:

1. The system topology is a non-hierarchical, regular, homogeneous, undirected graph of $N$ processor nodes, each with degree $d$.

2. Each process receives its own messages as well as those of its immediate neighbors. Thus, each process receives $d + 1$ messages in each voting round.

3. Messages received by a process are not relayed to another process. Thus, the physical and logical connectivity are identical.
4. $N \gg d$ so that the network diameter $k \gg 1$. Thus, “wrap-around” effects can not assist the local convergence process in a given voting round.

3.2 Definitions

The following sets and multisets describe the relationships between the values received by two processes $i$ and $j$ in a partially connected system.

\begin{align*}
P_i &= \text{The set of processes whose values are receivable by process } i. \ P_j \text{ is similarly defined for process } j. \\
P_{i \cap j} &= P_i \cap P_j, \text{ the set of processes whose values are receivable by both processes } i \text{ and } j. \\
P_{i \cup j} &= P_i \cup P_j, \text{ the set of processes whose values are receivable by either process } i, \text{ process } j, \text{ or both.} \\
U_{i \cap j} &= \text{The multiset of all values generated by non-faulty processes in } P_{i \cap j}. \\
U_{i \cup j} &= \text{The multiset of all values generated by non-faulty processes in } P_{i \cup j}. 
\end{align*}

In a completely connected system, $U_{i \cap j} = U_{i \cup j}$. However, in a partially connected system, $U_{i \cap j} \subseteq U_{i \cup j}$. We therefore define two types of local convergence for partially connected systems, Union Convergence and Intersection Convergence.

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

\begin{align*}
\text{U1: } & F(V_i) \in \rho(U_{i \cup j}), \text{ and } F(V_j) \in \rho(U_{i \cup j}), \\
\text{U2: } & |F(V_i) - F(V_j)| \leq C \delta(U_{i \cup j}), \text{ where } 0 \leq C < 1.
\end{align*}

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

\begin{align*}
\text{I1: } & F(V_i) \in \rho(U_{i \cap j}), \text{ and } F(V_j) \in \rho(U_{i \cap j}), \\
\text{I2: } & |F(V_i) - F(V_j)| \leq C \delta(U_{i \cap j}), \text{ where } 0 \leq C < 1.
\end{align*}
Parameter $C$ is the *Convergence Rate* of a voting algorithm, the primary measure of its effectiveness.

In fully connected networks, local convergence implies global convergence, since every node is connected to every other node. But this is not necessarily true in partially connected networks. In other words, while local convergence is a prerequisite to global convergence in partially connected networks, it does not guarantee single-step global convergence. More specifically, consider two immediate neighbors $i$ and $j$. For a given multiset, such as $U_{i,j}$, the conditions $U_1$ and $U_2$ are satisfied, reducing the range and diameter values. Although this is the property of single-step convergence, it is a special case of it, because of the interference of the neighboring processes. For instance, in a next round, 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, nodes $i$ 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, local convergence is a special case of the single-step convergence definition, and global convergence is *asymptotic* rather than monotonic [6].

### 3.3 Perspective

Convergent voting in partially connected systems differs from completely connected systems in two respects. The first difference is in the definitions of convergence. The relationships between local and global convergence and between Intersection and Union convergence are shown in Fig. 1. In a completely connected system, $P_{i\cap j} = P_{i\cup j}$ so that Union and Intersection Convergence are identical. Furthermore, for each process $i$, $P_i$ is the set of all processes so that local and global convergence are identical. Thus all four forms of convergence degenerate to a single form in completely connected systems.

The second major difference between completely connected and partially connected systems is their handling of benign faults. In a completely connected system, benign faults can be ignored because all processes can delete the benign errors from $V$ and vote with a smaller
sized multiset [6]. 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 [11]. If self-evident faults are discarded, different processes will use different sized voting multisets so that $V$ would not be identical for all processes. Therefore, in partially connected systems, only symmetric and asymmetric faults are considered. Any fault which would have qualified as benign in a completely connected system becomes symmetric in a partially connected system.

4 SYNCHRONOUS SYSTEMS

This section presents an overview of recent results [7] on the performance and fault tolerance of MSR algorithms in *synchronous* partially connected systems. These results serve as the basis for our subsequent analysis of MSR algorithms in *asynchronous* partially connected systems.

In a synchronous system there is a finite bound on the processing and communication delays of each round. A convergent voting algorithm expects to receive data from all non-faulty processes within this bound. An arbitrary default value is chosen for any data not received within the bound. Thus, at the time the voting algorithm starts, all non-faulty processes have the same number of elements $V$ in their voting multisets.
There are two parameters of primary interest. The first is the minimum value of \( V \) for which convergence can be guaranteed in a given fault scenario. The second is the maximum bound on convergence rate \( C \).

### 4.1 Fault-Tolerance

Before discussing fault-tolerance in the synchronous case, it is necessary to review some definitions from [7]:

\[
\begin{align*}
\alpha & = \text{The number of asymmetrically faulty processes in } P_{i \mid j}. \\
\beta & = \text{The number of symmetrically faulty processes in } P_{i \mid j}. \\
\chi & = |P_i| - |P_{i \mid j}| = |P_j| - |P_{i \mid j}|, \text{ the number of processes whose values are receivable by either } i \text{ or } j, \text{ but not by both.} \\
f & = \text{The maximum number of faults in either } \{P_i \setminus P_{i \mid j}\} \text{ or } \{P_j \setminus P_{i \mid j}\}, \text{ regardless of the fault modes exhibited.}
\end{align*}
\]

#### 4.1.1 Union Convergence

It is known that a synchronous MSR algorithm \( F(V) = \text{mean}[Sel_\sigma(\text{Red}^\tau(V))] \) can guarantee Union Convergence only if [7]:

\[
\begin{align*}
\tau & \geq (\alpha + f) + s, \quad (4.1) \\
\sigma & \begin{cases} 
\geq 1 & \text{if } |a + \chi| = 0 \\
\geq 2 & \text{if } |a + \chi| > 0
\end{cases}, \quad (4.2) \\
V & \geq 2\tau + max(|a + \chi| + 1, \sigma). \quad (4.3)
\end{align*}
\]

Substituting the minimum values of \( \tau \) and \( \sigma \) from (4.1) and (4.2) into (4.3) shows that a Union Convergent MSR voting algorithm can exist only if:

\[
V \geq (3\alpha + 2s + 1) + (\chi + 2f). \quad (4.4)
\]

The minimum values of \( P_i \) and the degree \( d \) which permit Union Convergence between two adjacent processes can be derived by noting that \( d = |P_i| - 1 = V - 1 \). Then, using (4.4)
convergence can be guaranteed only if:

\[
d \geq (3a + 2s) + (\chi + 2f). \tag{4.5}
\]

In addition, non-adjacent processes can be convergent if their intersection set is large enough. Recall that \(|P_{i \cap j}| = V - \chi\). Thus, by (4.4), Union Convergence between processes \(i\) and \(j\) can be guaranteed only if:

\[
|P_{i \cap j}| \geq (3a + 2s + 1) + (2f). \tag{4.6}
\]

### 4.1.2 Intersection Convergence

It is known that a synchronous MSR algorithm \(F(V) = \text{mean} [Sel_{\sigma} (Red^T (V))]\) can guarantee Intersection Convergence only if [7]:

\[
\tau \geq (a + \chi) + s, \tag{4.7}
\]

\[
\sigma \begin{cases}
\geq 1 & : |a + \chi| = 0 \smallskip \\
\geq 2 & : |a + \chi| > 0
\end{cases}, \tag{4.8}
\]

\[
V \geq 2\tau + \max (|a + \chi| + 1, \sigma). \tag{4.9}
\]

Substituting the minimum values for \(\sigma\) and \(\tau\) from (4.7) and (4.8) into (4.9) shows that an Intersection Convergent MSR algorithm can exist only if:

\[
V \geq (3a + 2s + 1) + (3\chi). \tag{4.10}
\]

Thus, for Intersection Convergence between processes \(i\) and \(j\), (4.10) requires:

\[
d = V - 1 \geq (3a + 2s) + (3\chi), \smallskip \\
|P_{i \cap j}| = V - \chi \geq (3a + 2s + 1) + (2\chi).
\]

### 4.2 Convergence Rate

An important result from [7] is the ease with which convergence rate \(C\) can be determined for any MSR voting algorithm. Specifically, for both Intersection and Union Convergence:

\[
C = \frac{\gamma_{a+\chi}}{\sigma}, \tag{4.11}
\]
where $\sigma$ is the size of the selected multiset $S$, and $\gamma_{a+\chi}$ is defined below$^3$. Parameter $\gamma_{a+\chi}$ depends on the selection function $Sel_\sigma$ employed in a particular voting algorithm. To define $\gamma_{a+\chi}$, we first consider how the elements of $S$ are distributed within $M$. By the definition of an MSR algorithm, the selected multiset $S = \langle s_1, \ldots, s_\sigma \rangle$ is a subsequence of the medial multiset $M = \langle m_1, \ldots, m_M \rangle$. Let $g$ be an index into $S$. Then, for each $g \in \{1, \ldots, \sigma\}$ there exists exactly one $k(g) \in \{1, \ldots, M\}$ which guarantees that $s_g = m_{k(g)}$ for all possible $M$. Given two indices $g, h \in \{1, \ldots, \sigma\}$ where $g \leq h$, we define $\Delta k(g, h)$ as the number of elements in $M$ spanned by elements $\langle s_g, \ldots, s_h \rangle$ in $S$. Specifically:

$$\Delta k(g, h) = \lfloor k(h) - k(g) \rfloor.$$

(4.12)

Now for a given non-negative integer $\alpha$, define $\gamma_\alpha$ as the minimum value of $(h - g)$ for which $\Delta k(g, h) \geq \alpha$. By definition, $\gamma_\alpha$ can not be negative. Thus, $\gamma_\alpha$ does not exist if it is negative at the end of the pseudo-code.

It has been shown [7] that if $\gamma_{a+\chi}$ exists for a particular synchronous MSR algorithm, then that algorithm is convergent, and the convergence rate $C$ is given by (4.11). Conversely, if $\gamma_{a+\chi}$ does not exist, then at least one fault scenario exists in which the algorithm is non-convergent.

5 ASYNCHRONOUS SYSTEMS

In asynchronous systems, as in synchronous systems, each non-faulty process will transmit a value for each voting round. However, asynchronous systems impose no finite bounds on process operation or communication delays. There is thus no fixed time at which a process is guaranteed to have received a value from all non-faulty processes. Each process must decide for itself when to stop waiting for messages and begin the voting algorithm for the current round [5].

In an asynchronous MSR algorithm, non-faulty process $i$ votes on a new value when it receives

$^3$In [7], the notation $\gamma'$ is used instead of $\gamma_{a+\chi}$. This new notation is adopted to ensure consistency with recent publications.
the first $|P_i| - \tau$ values. If $\tau$ is not less than the maximum number of faulty processes in $P_i$, then there are at least $|P_i| - \tau$ non-faulty processes participating in each round. Each non-faulty process is required to eventually send one value for each round. Thus, each non-faulty process $i$ will eventually receive $V = |P_i| - \tau$ values (including its own) and start its voting algorithm.

Simple expressions will now be derived for the convergence rate and fault-tolerance of any MSR voting algorithm in an asynchronous partially connected system. It will be shown that these expressions can be derived as a variant of synchronous convergence.

5.1 Union Convergence

THEOREM 1 : Given non-negative integers $a$, $s$, $f$, $\chi$, and $\tau$, multisets $V_i$ and $V_j$ of size $V$, and an MSR algorithm $F(V) = \text{mean} [Sel_\sigma (\text{Red}_\tau (V))]$, then the algorithm can be union convergent only if:

\[
\begin{align*}
\tau & \geq (a + s) + f, \\
\sigma \begin{cases} 
\geq 1 & : [a + \tau + \chi] = 0 \\
\geq 2 & : [a + \tau + \chi] > 0 \end{cases}, \\
V & \geq 2\tau + \max (a + \tau + \chi + 1, \sigma).
\end{align*}
\]

Furthermore, if $\gamma_{a+\tau+\chi}$ exists, then the algorithm is convergent with convergence rate:

\[
C = \frac{\gamma_{a+\tau+\chi}}{\sigma}.
\]

PROOF : The maximum number of faulty processes in $P_i$ is $a + s + f$. If $\tau < a + s + f$, then there may be an extremal error remaining in the medial multiset $\text{Red}_\tau (V_i)$. This value could skew the mean such that the voted value is outside of $\rho(U_{i;j})$. Thus, (1) is necessary to assure condition $|U_1|$ for Union Convergence.
For a particular round of voting, let $X_i$ be the set of processes whose values are received by process $i$ but not by process $j$. Similarly, let $X_j$ be the set of processes whose values are received by $j$ but not by $i$. We now observe:

1. In both synchronous and asynchronous systems, $X_i$ includes the $\chi$ processes in $\{P_i \backslash P_{irj}\}$, and $X_j$ includes the $\chi$ processes in $\{P_j \backslash P_{irj}\}$.

2. In asynchronous systems process $i$ waits for $|P_i| - \tau$ values before voting. Due to the indeterminacy in message delivery delays, the $\tau$ values not received by $i$ may have originated from within $P_{irj}$. Thus, $X_j$ may include up to $\tau$ processes from $P_{irj}$ whose values are not received by $i$. Similarly, $X_i$ may include up to $\tau$ processes from $P_{irj}$ whose values are not received by $j$.

Based on these two observations, we can conclude:

$$|X_i| = |X_j| \leq (\chi) \quad \text{in synchronous systems,}$$

(5)

$$|X_i| = |X_j| \leq (\tau + \chi) \quad \text{in asynchronous systems.}$$

(6)

In the worst case, $X_i$ and $X_j$ are disjoint, $(X_i \cap X_j = \emptyset)$, so that the inequalities in (5) and (6) become equalities. In this case, for each process $x_i \in X_i$ there is a corresponding process $x_j \in X_j$, such that $x_i \neq x_j$. A given pair $(x_i, x_j)$ may transmit different values to processes $i$ and $j$, respectively. Thus, the pair $(x_i, x_j)$ can exhibit the same behavior as a single asymmetrically faulty process, regardless of the health of $x_i$ and $x_j$. Thus, the effective number of asymmetric faults is $a + |X_i|$.

For synchronous systems, (5) shows that there are in the worst case $(\chi)$ processes in $X_i$. Thus, the effective number of asymmetric faults is $(a + \chi)$.

For asynchronous systems, (6) shows that there are in the worst case $(\tau + \chi)$ processes in $X_i$. In other words, the $\tau$ processes in $X_i$ from $P_{irj}$ mimic the behavior of the $\chi$ processes from $P_i \backslash P_{irj}$. Thus, up to $\tau + \chi$ processes in the asynchronous case act like $\chi$ processes in the synchronous case. Therefore, the effective number of asymmetric faults is $(a + \tau + \chi)$ rather than $(a + \chi)$.
Thus, for any two processes $i$ and $j$, the results of [7] as stated in (4.2), (4.3), and (4.11) are valid for asynchronous Union Convergent systems if $a + \tau$ is substituted for $a$, producing (2), (3), (4) above. □

To determine the minimum conditions for the existence of an asynchronous Union Convergent MSR voting algorithm, we apply the minimum values for $\sigma$ and $\tau$ to the minimum value of $V$ in Theorem 1.

$$V \geq 2\tau + \max(a + \tau + \chi + 1, \sigma)$$

$$= 2\tau + a + \tau + \chi + 1$$

$$= 3\tau + a + \chi + 1$$

$$\geq 3(a + s + f) + a + \chi + 1$$

$$= (4a + 3s + 1) + (\chi + 3f). \quad (5.1)$$

Since an asynchronous algorithm waits for $|P_i| - \tau$ values before voting, $|P_i| = V + \tau$. Thus, from (5.1) the minimum number of processes in $P_i$ is:

$$|P_i| = V + \tau$$

$$\geq (4a + 3s + 1) + (\chi + 3f) + \tau$$

$$\geq (4a + 3s + 1) + (\chi + 3f) + (a + s + f)$$

$$= (5a + 4s + 1) + (\chi + 4f).$$

Since each process receives its own messages, the minimum required degree of the network is:

$$d = |P_i| - 1 \geq (5a + 4s) + (\chi + 4f). \quad (5.2)$$

Furthermore, the minimum required size of the intersection set is:

$$|P_{i\cap j}| = |P_i| - \chi \geq (5a + 4s + 1) + (4f). \quad (5.3)$$

5.2 Intersection Convergence

In a partially connected regular system, $P_{i\cap j} \subseteq P_{i\cup j}$. Thus, Intersection Convergence is a more stringent convergence criterion than Union Convergence since it demands convergence
within a smaller multiset.

**Theorem 2**: Given non-negative integers \( a, s, \chi, \) and \( \tau, \) multisets \( \mathbf{V}_i \) and \( \mathbf{V}_j \) of size \( V \) and an MSR algorithm \( F(\mathbf{V}) = \text{mean}[\text{Sel}_\sigma(\text{Red}^\tau(\mathbf{V}))], \) then the algorithm can be intersection convergent only if:

\[
\begin{align*}
\tau & \geq (a + s) + \chi, \quad (1) \\
\sigma & \geq 1 \quad : \quad [a + \tau + \chi] = 0 \quad , \quad (2) \\
\sigma & \geq 2 \quad : \quad [a + \tau + \chi] > 0 \quad , \quad (3)
\end{align*}
\]

Furthermore, if \( \gamma_{a+s+\chi} \) exists, then the algorithm is convergent with convergence rate:

\[
C = \frac{\gamma_{a+s+\chi}}{\sigma}. \quad (4)
\]

**Proof**: Any faulty process can transmit a value outside of \( \rho(U_{i\cap j}) \). In addition, any process in \( \{P_i \setminus P_{i\cap j}\} \) or \( \{P_j \setminus P_{i\cap j}\} \) could also transmit a value outside of \( \rho(U_{i\cap j}) \) even if it is non-faulty. Thus, the maximum number of processes in \( P_i \) which could transmit a value outside of \( \rho(U_{i\cap j}) \) is \( a + s + \chi \). If \( \tau < a + s + \chi \), then there may be an extremal value, outside of \( \rho(U_{i\cap j}) \), remaining in the medial multiset \( \text{Red}^\tau(\mathbf{V}_i) \). This value could skew the mean such that the voted value is outside of \( \rho(U_{i\cap j}) \). Thus, \( (1) \) is necessary to meet condition \([I1]\) for Intersection Convergence.

It is now necessary to show that the effective number of asymmetric faults is \( (a + \tau + \chi) \) as opposed to \( (a + \chi) \) so that \( (2), (3), \) and \( (4) \) above follow from \( (4.8), (4.9), \) and \( (4.11) \). The proof of this case is essentially identical to the proof of Theorem 1. □

To determine the minimum conditions for the existence of an asynchronous Intersection Convergent MSR voting algorithm, we apply the minimum values for \( \sigma \) and \( \tau \) to the minimum
value of $V$ in Theorem 2:

$$V \geq 2\tau + \max(a + \tau + \chi + 1, \sigma)$$

$$= 2\tau + a + \tau + \chi + 1$$

$$= 3\tau + a + \chi + 1$$

$$\geq 3(a + s + \chi) + a + \chi + 1$$

$$= (4a + 3s + 1) + 4\chi$$

We can now determine the minimum values of $|P_i|$, $d$ and $|P_{i\cap j}|$ required for asynchronous Intersection Convergence between processes $i$ and $j$:

$$|P_i| = V + \tau$$

$$\geq (4a + 3s + 1) + 4\chi + \tau$$

$$\geq (4a + 3s + 1) + 4\chi + (a + s + \chi)$$

$$= (5a + 4s + 1) + 5\chi$$

$$d = |P_i| - 1 \geq (5a + 4s) + (5\chi)$$

$$|P_{i\cap j}| = |P_i| - \chi \geq (5a + 4s + 1) + (4\chi)$$

5.3 Network Topologies Example

Table 1 summarizes the relevant minimum parameters for Intersection and Union Convergence in both synchronous and asynchronous systems. This table clearly shows that for a given number of faults, asynchronous systems require denser interconnections than synchronous systems. Hence, asynchronous systems are less fault-tolerant than synchronous systems.

To demonstrate the results in Table 1, we consider three common mesh topologies as shown in Fig. 2. These meshes have degrees of $d = 4$, $d = 6$, and $d = 8$, respectively. For each mesh, two nodes have been labeled $i$ and $j$ such that $|P_{i\cap j}|$ is maximized. Nodes enclosed
Table 1: Summary of Minimum Convergence Parameters

<table>
<thead>
<tr>
<th>TYPE</th>
<th>PARAM</th>
<th>SYNCHRONOUS</th>
<th>ASYNCHRONOUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNION</td>
<td>$\tau \geq \frac{2a + f}{3} + 2$</td>
<td>$a + s + f$</td>
<td>$a + s + f$</td>
</tr>
<tr>
<td></td>
<td>$d \geq \frac{3a + 2s + 1}{2} + f$</td>
<td>$(3a + 2s + 1) + (2f)$</td>
<td>$(5a + 4s + 1) + (4f)$</td>
</tr>
<tr>
<td>INTERSECTION</td>
<td>$</td>
<td>P_{i\cap j}</td>
<td>\geq \frac{3a + 2s + 1}{2}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$(3a + 2s + 1) + (2\chi)$</td>
<td>$(5a + 4s + 1) + (4\chi)$</td>
</tr>
</tbody>
</table>

within a dashed box comprise $P_{i\cap j}$ for that mesh. Inspection of Fig. 2 reveals that in all three meshes $\chi = 3$ for nodes $i$ and $j$.

![Figure 2: Common Mesh Networks](image)

The values of $d$, $\chi$ and $P_{i\cap j}$ for each mesh can be substituted into the equations in Table 1 to determine the robustness of each topology to any given fault scenario. The result is that with $\chi = 3$, Intersection Convergence requires $d \geq 9$ even in a fault-free synchronous system. Thus, none of these networks is Intersection Convergent.

Table 2 shows the Union Convergence properties of the three meshes in both synchronous and asynchronous systems. Results have been calculated for all scenarios containing up to two faults. As expected from the equations in Table 1, synchronous systems are more robust
than asynchronous systems, and the more densely connected networks are more robust than the less densely connected networks.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a s f</td>
<td>Async</td>
<td>Sync</td>
<td>Async</td>
</tr>
<tr>
<td>0 0 0</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>0 0 1</td>
<td></td>
<td></td>
<td>U</td>
</tr>
<tr>
<td>0 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 2 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Key: U ⇒ Union Convergent, blank ⇒ Non-Convergent.

6 CONCLUSIONS

This paper has addressed fault-tolerant convergent voting in asynchronous partially connected systems. It has analyzed convergence rates and fault tolerance for a broad family of "locally" convergent voting algorithms. In our analysis, the relay of messages is prohibited so that each processor communicates only with its immediate neighbors. This constraint allows convergent voting in large sparsely connected systems where communication overhead makes large-scale message relaying impractical.

The convergence rate and fault-tolerance are easily determined for any algorithm in the MSR family. Thus, a system designer can quickly find the appropriate voting algorithm to match the needs of a specific system. Numerical results showed that for a given number of faults, asynchronous systems require denser interconnections than synchronous systems. Hence, asynchronous systems are less fault-tolerant than synchronous systems.
It was shown that asynchronous partially connected systems can be treated as a variant of the synchronous case described in [7]. Specifically, to obtain the convergence rate and fault-tolerance, it was necessary to determine the effective number of asymmetric faults as seen by two processes. The effective number of asymmetric faults in synchronous convergence is \((a + \chi)\), whereas that of asynchronous convergence is \((a + \tau + \chi)\). By substituting \(a + \tau\) for \(a\) in the asynchronous case, the problem was treated like the synchronous case. The value of \(a + \tau + \chi\) was also used to define \(\gamma_{a+\tau+\chi}\) as the appropriate spanning parameter. The resulting convergence rate was shown to be \(C = \gamma_{a+\tau+\chi}/\sigma\).

In many system applications, the goal of convergent voting is to achieve Approximate Agreement on a global level, i.e. to reduce the range of values held by all non-faulty processes. It is known that local Union Convergence is a prerequisite to global convergence. The quantitative relationship between local convergence rate and global convergence rate is currently being studied in the context of synchronous systems [7, 20, 21]. At this time, it appears that global convergence properties are independent of whether local convergence is achieved by synchronous or asynchronous algorithms.

Global convergence in partially connected network under different hybrid fault models is very complex due to many difficulties such as asymptotic convergence rather than single-step convergence, the fact that convergence is highly dependent on the initial distribution of values, and the availability of different topologies as opposed to a single topology for fully connected networks. Nevertheless, some progress has been made. For instance, in [20], the global convergence has been obtained for a hexagonal mesh that can tolerate omission faults. In a different study, global convergence is being obtained for a form of partially connected network with limited relays which can take advantage of the already available results for fully connected networks. The so-far results have shown improvement in communication complexity and link complexity of the network.
References


