The Weak Byzantine Generals Problem

L. LAMPORT

SRI International, Menlo Park, California

Abstract The Byzantine Generals Problem requires processes to reach agreement upon a value even though some of them may fail. It is weakened by allowing them to agree upon an "incorrect" value if a failure occurs. The transaction commit problem for a distributed database is a special case of the weaker problem. It is shown that, like the original Byzantine Generals Problem, the weak version can be solved only if fewer than one-third of the processes may fail. Unlike the original problem, an approximate solution exists that can tolerate arbitrarily many failures.

Categories and Subject Descriptors: D.4.5 [**Operating Systems**]: Reliability, F.2.m [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms: Reliability

Additional Key Words and Phrases: Agreement, interactive consistency, distributed systems

1. *Introduction*

The Byzantine Generals Problem involves obtaining agreement among a collection of processes, some of which may be faulty. It can be stated precisely as follows.

Byzantine Generals Problem: Given a collection of processes numbered from 0 to $n - 1$ which communicate by sending messages to one another, to find an algorithm by which Process 0 can transmit a value v to all the processes such that:

(1) If Process 0 is nonfaulty, then any nonfaulty Process i obtains the value v.
(2) If Processes i and j are nonfaulty, then they both obtain the same value.

Note that condition 2 follows from condition 1 if Process 0 is nonfaulty.

Nonfaulty processes are assumed to correctly follow their algorithm, but faulty processes may do anything. We assume that the absence of a message is detectable, which is equivalent to assuming that a faulty process sends every message that it is supposed to—although it need not send the *correct* message. The difficulty of the problem lies in the fact that a faulty process may send conflicting information to two different processes.

This problem was described in [1] in terms of a Byzantine general metaphor—hence its name. Essentially the same problem appeared in [2], where it was called the Interactive Consistency Problem. The problem was shown there to be solvable if and only if fewer than one-third of the processes are faulty—unless unforgeable, signed

This work was supported in part by the National Science Foundation under Grant MCS 81-04459.

Author's address: Computer Science Laboratory, SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025

messages are assumed. In particular, no solution works for three processes in the presence of a single fault.

In this paper, we consider a weaker version of the problem, in which condition 1 is replaced by the following:

(1) If all processes are nonfaulty, then every Process i obtains the value v.

The transaction commit problem for a distributed database is an instance of this weaker problem, in which Process 0 represents a transaction coordinator, and the other processes represent the database sites affected by the transaction [3]. The commit coordinator's "value" is its decision of whether to commit or abort the transaction. All sites must agree on whether the transaction is committed or aborted (condition 2), but the failure of any site is allowed to abort the transaction—hence the weaker version of condition 1.

Any solution to the original Byzantine Generals Problem is obviously a solution to the Weak Byzantine Generals (WBG) Problem, so the WBG Problem is solvable if fewer than one-third of the processes may be faulty. In Section 2, we prove the converse: no solution exists if one-third or more of the processes are faulty. Hence, the WBG Problem is solvable in precisely those situations in which the original Byzantine Generals Problem is. However, in Section 3 we give a "solution" that works with any number of faulty processes, but requires the processes to send an infinite number of messages before choosing their values. Of course, this "solution" is of no practical interest, since it cannot be implemented. Its interest lies in the fact that the original Byzantine Generals Problem does not possess such a "solution". (The impossibility proof of [2] did not assume a finite number of messages.) Hence, the WBG Problem is in some sense strictly weaker than the Byzantine Generals Problem.

In Section 3, we also show that if condition 2 of the WBG Problem is replaced by a weaker condition requiring only approximate equality, then the problem is solvable with any number of faulty processes. More precisely, if the set of possible values is a bounded set of numbers, then for any $\epsilon > 0$ there is an algorithm which guarantees that the values chosen by any two nonfaulty processes differ by less than ϵ. It was shown in [1] that no such approximate solution exists for the original Byzantine Generals Problem.

The Byzantine Generals Problem arises in practice when trying to get the nonfaulty processes to agree upon the value of some input quantity. As discussed in [1], it is central to the implementation of fault-tolerant computer systems. The WBG Problem arises when trying to get the nonfaulty processes simply to agree, regardless of what they agree upon. To eliminate the trivial possibility of having them agree upon a prearranged value, we can assume that each process chooses a private value, and that these private values are used in reaching agreement upon a single public value. The general problem of reaching agreement can then be formulated as follows:

Weak Interactive Consistency Problem: Each Process i chooses a private value w_i. The processes must then communicate among themselves to allow each process to compute a public value, such that:

(1) If all processes are nonfaulty and all the w_i have the same value, then every process computes this value as its public value.
(2) Any two nonfaulty processes compute the same public value.

It is easy to show that this is equivalent to the WBG Problem. First of all, it is easy to see that if Process 0 transmits the value w_0 to all processes using a solution to the

WBG problem, and all nonfaulty processes choose the value they obtain as their public value, then the above two conditions hold, so this is a solution to the Weak Interactive Consistency Problem. Conversely, given a solution to the Weak Interactive Consistency Problem, a solution to the WBG problem is obtained by having Process 0 send its value v to all the processes, and then letting each Process i use the value it received as its private value in the Weak Interactive Consistency solution.

2. *Impossibility Result*

In this section, we prove that no solution to the WBG problem exists if one-third or more of the processes are faulty. This requires a precise statement of what constitutes a WBG solution. We begin with some notation. (A glossary of all our notation appears at the end of the paper.) Let P denote the set $\{0, \ldots, n-1\}$, and P^* the set of all finite sequences of elements of P (including the null sequence). Let Π denote the set of all finite sequences of the form $0, \pi$ with $\pi \in P^*$—i.e., all elements of P^* whose first element is 0. We think of $0, p_1, \ldots, p_k$ as the path of length k traveled by a message that starts at Process 0 and is relayed via Processes $p_1, \ldots, p_{k-1}$ to Process p_k. (This is different from the notation used in [2].) Let Π_i denote the subset of Π consisting of all sequences ending in i—i.e., all message paths leading from Process 0 to Process i.

A *scenario* Φ is a mapping from Π into a set of values V. If we think of an element π of Π as a message path, then $\Phi(\pi)$ is the contents of the message received at its final destination. We say that Process i is *nonfaulty* in a scenario Φ if for every message path π, $i \in \Pi$ and every $j \in P$: $\Phi(\pi, i, j) = \Phi(\pi, i)$. In other words, i is nonfaulty in Φ if Process i correctly relays all messages. If all processes are nonfaulty in Φ, then $\Phi(\pi) = \Phi(0)$ for all $\pi \in \Pi$.

We define an *i-scenario* to be a mapping from Π_i to V. An i-scenario thus describes the contents of the messages received by Process i. For any scenario Φ, we let Φ_i be the i-scenario that is the restriction of Φ to Π_i, so Φ_i is the part of Φ "seen" by Process i.

A solution to the WBG problem consists of an algorithm by which the processes send messages to one another based upon the contents of messages already received. Initially, the only information is the value v, which is known only to Process 0. Therefore, all information travels along paths in Π.[1] To send the maximum amount of information to one another, Process 0 would send the value v to all processes, and then processes would send one another the contents of every message they receive. Thus, if $\Phi(0)$ equals v, then a scenario Φ describes the maximum amount of information that the processes could send to one another. A nonfaulty process can always ignore information that it receives, and a faulty process can do anything—including guess any information that was withheld from it. Hence, any algorithm for choosing values based upon information sent among the processes can be described in terms of an algorithm based upon the entire scenario Φ. We therefore make the following definition.

Definition. An *m-fault* WBG Algorithm B consists of a set of mappings B_i from i-scenarios into V, for all $i \in P$, such that for any scenario Φ in which at least $n - m$

[1] We could have considered paths starting from other processes than Process 0 as well, and the impossibility proof would remain essentially the same. However, for simplicity we have restricted ourselves to message paths in Π

processes are nonfaulty:

(1) If all processes in P are nonfaulty in Φ, then for all $i \in P$: $B_i(\Phi_i) = \Phi(0)$.
(2) For any $i, j \in P$: if i and j are nonfaulty in Φ, then $B_i(\Phi_i) = B_j(\Phi_j)$.

We will show that no m-fault WBG algorithm exists if $3 \leq n \leq 3m$. (The problem becomes trivial if $n \leq 2$.)

If the value of $B_i(\Phi_i)$ depended upon the entire infinite i-scenario Φ_i, then the algorithm B would require an infinite amount of message passing and would not be a real solution to the WBG Problem. We thus make the following definition, where $\Pi^{(k)}$ is defined to be the set of message paths in Π of length at most k, and $\Pi_i^{(k)} = \Pi^{(k)} \cap \Pi_i$.

Definition. A WBG algorithm B is said to be *finite* if for every scenario Φ there is an integer k such that for any scenario Ψ and all $i \in P$: if the restrictions of Φ_i and Ψ_i to $\Pi^{(k)}$ are equal, then $B_i(\Phi) = B_i(\Psi)$.

A finite WBG algorithm is one in which for every scenario, there is a k such that each process can choose its value after k rounds of message passing. This is a natural definition, since it insures that every process is eventually able to choose a value. However, it does not immediately rule out the possibility that the required number of rounds k can become arbitrarily large. We now show that this is not the case, and that a single value of k can be chosen for all scenarios.

LEMMA 1. *For any finite WBG algorithm B there is a nonnegative integer k such that for any scenarios Φ and Ψ and all $i \in P$: if the restrictions of Φ_i and Ψ_i to $\Pi_i^{(k)}$ are equal, then $B_i(\Phi_i) = B_i(\Psi_i)$.*

PROOF. Define an r-level finite scenario to be a mapping from $\Pi^{(r)}$ to V. For any fixed i, we define a tree structure on the set of all such finite scenarios by letting an r-level scenario Φ be an ancestor of an r'-level scenario Φ' if $r < r'$ and Φ_i is the restriction of Φ'_i to $\Pi_i^{(r)}$. Consider the subtree consisting of r-level scenarios Φ, for all r, such that there exist (infinite) scenarios Ψ and Ω whose restrictions to $\Pi_i^{(r)}$ equal Φ_i, and for which $B_i(\Psi_i)$ does not equal $B_i(\Omega_i)$. If this subtree were infinite, then by Konig's lemma it would have an infinite path. Such an infinite path defines an infinite scenario Φ which contradicts the definition of finiteness. Hence, this subtree must be finite, which implies the existence of a k_i such that for any scenarios Φ and Ψ: if the restrictions of Φ_i and Ψ_i to $\Pi_i^{(k_i)}$ are equal, then $B_i(\Phi_i) = B_i(\Psi_i)$. To complete the proof, we let k equal $sup\{k_i : i \in P\}$. □

To prove the nonexistence of an m-fault algorithm when $n \leq 3m$, we first prove the nonexistence of a 1-fault algorithm for $n = 3$. Therefore, until further notice, we assume that $P = \{0, 1, 2\}$.

We define the signed distance function δ on P by:

$$\delta(0, 1) = \delta(1, 2) = \delta(2, 0) = 1,$$
$$\delta(i, j) = -\delta(j, i).$$

For any path $\pi = 0, p_1, \ldots, p_k$ we define $\sigma(\pi)$ to equal

$$\sum_{i=1}^{k} \delta(p_{i-1}, p_i).$$

If we think of the processes 0, 1 and 2 being arranged clockwise in a circle, then $\delta(i, j)$ is the clockwise angular distance from i to j (where a distance of 3 represents a full circle), and $\sigma(\pi)$ is the signed angular distance traveled by the path π.

LEMMA 2. *For any path* $0, p_1, \ldots, p_k \in \Pi$: $\sigma(0, p_1, \ldots, p_k)$ *mod* $3 = p_k$.

PROOF. This is a simple consequence of the observation that

$$\delta(r, s) + \delta(s, t) \equiv \delta(r, t) \text{ mod } 3. \qquad \square$$

For any integer r, we let $\bar{r}$ denote r *mod* 3, which equals 0, 1, or 2.

We now choose two particular elements of V, which we denote T and F. The following lemma asserts the existence of a sequence of scenarios $\Phi^{(r)}$ for integral values of r (including negative integers) which will form the basis for a proof by contradiction. Only two values, denoted T and F, appear in $\Phi^{(r)}$. In this scenario, Processes $\overline{r+1}$ and $\overline{r+2}$ are nonfaulty, so they relay values correctly. The faulty Process $\bar{r}$ acts correctly except when relaying messages π for which $\sigma(\pi) = \bar{r}$, in which case it sends the value T to Process $\overline{r+1}$ and the value F to Process $\overline{r-1} = \overline{r+2}$.

LEMMA 3. *For any values* T *and* F *in* V, *and any integer* r *there is a scenario* $\Phi^{(r)}$ *such that for* $i = 1, 2$:

(1) *Process* $\overline{r+i}$ *is nonfaulty in* $\Phi^{(r)}$.

(2) *For any* $\pi \in \Pi_{\overline{r+i}}$:

$$\Phi^{(r)}(\pi) = \begin{cases} F & \text{if } \sigma(\pi) \geq r + i, \\ T & \text{if } \sigma(\pi) < r + i. \end{cases}$$

PROOF. By Lemma 2, condition 2 defines $\Phi^{(r)}_{\overline{r+i}}$ for $i = 1, 2$. Since there are no requirements on $\Phi^{(r)}_{\bar{r}}$, and Process $\bar{r}$ is allowed to be faulty, we need only show that Condition 2 is achievable when Processes $\overline{r+1}$ and $\overline{r+2}$ correctly relay messages to one another. However, this follows easily from the observation that if $\pi \in \Pi_{\overline{r+i}}$, then $\sigma(\pi, r + i \pm 1) = \sigma(\pi) \pm 1$. $\square$

Note that the two conditions of Lemma 3 define the values of all messages in the scenario $\Phi^{(r)}$ except for the ones that Process r sends to itself.

The following result is a simple corollary of Lemma 3.

LEMMA 4. *For any integer* r: *if* $\Phi^{(r)}$ *is as in Lemma* 3, *then* $\Phi^{(r)}_{\overline{r+2}} = \Phi^{(r+1)}_{\overline{r+2}}$.

We can now prove the impossibility of a 1-fault WBG algorithm with three processes.

LEMMA 5. *If there are at least two distinct elements in* V, *then there does not exist a* 1*-fault WBG algorithm for* $n = 3$.

PROOF. Let B be such an algorithm, and let T and F be distinct elements of V. Let Φ^T and Φ^F be the scenarios defined by $\Phi^T(\pi) = T$ and $\Phi^F(\pi) = F$ for all $\pi \in \Pi$. It follows from condition 1 of the definition of a WBG algorithm that $B_i(\Phi^T_i) = T$ and $B_i(\Phi^F_i) = F$ for all i. For each integer r, let $\Phi^{(r)}$ be the scenario whose existence was proved in Lemma 3.

Let k be the nonnegative integer whose existence is guaranteed by Lemma 1, with Φ^T substituted for Φ. Since $\sigma(\pi)$ is less than or equal to the length of π, for any π in $\Pi^{(k)}_{\overline{k+1}}$, we have $\sigma(\pi) \leq k < k + 1$, so $\Phi^{(k)}(\pi) = T$. Hence, the restrictions of the scenarios $\Phi^T_{\overline{k+1}}$ and $\Phi^{(k)}_{\overline{k+1}}$ to $\Pi^{(k)}_{\overline{k+1}}$ are equal, so we must have $B_{\overline{k+1}}(\Phi^{(k)}_{\overline{k+1}}) = T$. Similarly, choosing such a nonnegative integer k' for Φ^F, since $-\sigma(\pi)$ is less than or equal to the length of π, for any π in $\Pi^{(k')}_{\overline{-k'}}$ we have $\sigma(\pi) \geq -k' = (-k' - 1) + 1$, so $\Phi^{(-k'-1)}_{\overline{-k'}}(\pi) = F$. Hence, the restrictions of $\Phi^F_{\overline{-k'}}$ and $\Phi^{(-k'-1)}_{\overline{-k'}}$ to $\Pi^{(k')}_{\overline{-k'}}$ are equal, so $B_{\overline{-k'}}(\Phi^{(-k'-1)}_{\overline{-k'}}) = F$.

It follows from Lemma 4 that for any r: $B_{r+2}(\Phi_{r+2}^{(r)}) = B_{r+2}(\Phi_{r+2}^{(r+1)})$. Since $r+1$ and $r+2$ are nonfaulty in $\Phi^{(r)}$, it follows from condition 2 of the definition of a WBG algorithm that $B_{r+1}(\Phi_{r+1}^{(r)}) = B_{r+2}(\Phi_{r+2}^{(r)})$. Hence, for each r: $B_{r+1}(\Phi_{r+1}^{(r)}) = B_{r+2}(\Phi_{r+2}^{(r+1)})$. A simple induction argument then shows that $B_{k+1}(\Phi_{k+1}^{(k)}) = B_{-k'}(\Phi_{-k'}^{(-k'-1)})$. However, we saw above that $B_{k+1}(\Phi_{k+1}^{(k)}) = T$ and $B_{-k'}(\Phi_{-k'}^{(-k'-1)}) = F$. Since T and F are distinct elements, this provides the required contradiction. □

We can now prove our main result.

THEOREM I. *If $n > 2$ and V contains at least two distinct elements, then there exists an m-fault WBG algorithm if and only if $n > 3m$.*

PROOF. The "if" part follows from the existence of algorithms to solve the original Byzantine Generals Problem, demonstrated in [1] and [2]. To prove the "only if" part, we assume the existence of such an algorithm and derive a contradiction.

Assume B is an m-fault WBG algorithm, with $3 \leq n \leq 3m$. We will use it to construct a 1-fault WBG algorithm for three processes, thereby contradicting Lemma 5. We first partition the (n-element) set P into three nonempty, disjoint sets P_0, P_1, P_2 each containing at most m elements. (We can do this because $3 \leq n \leq 3m$.) Let 0 be an element of P_0. We define the mapping λ: $P \to \{0', 1', 2'\}$ by letting $\lambda(p) = i'$ if and only if $p \in P_i$. We extend λ to a mapping from P^* into $\{0', 1', 2'\}^*$ in the obvious way by letting $\lambda(p_0, \ldots, p_k) = \lambda(p_0), \ldots, \lambda(p_k)$. We also let $0''$, $1''$, $2''$ be elements in P such that $0'' = 0$, $1'' \in P_1$ and $2'' \in P_2$. Hence, $\lambda(i'') = i'$.

We construct a 1-fault WBG algorithm B' for the set $P' = \{0', 1', 2'\}$ as follows. For any scenario Φ' on P', we define the scenario $\Lambda[\Phi']$ on P by $\Lambda[\Phi'](\pi) = \Phi'(\lambda(\pi))$. The WBG algorithm B' is defined by $B'_{i'}(\Phi'_{i'}) = B_{i''}(\Lambda[\Phi'_{i''}])$. Observe that if i' is nonfaulty in Φ', then every process in P_i (including i'') is nonfaulty in $\Lambda[\Phi']$.

To show that B' is a 1-fault WBG algorithm, we must verify the following two conditions.

(1) If all processes in P' are nonfaulty in Φ', then for all $i' \in P'$: $B'_{i'}(\Phi'_{i'}) = \Phi'(0')$.
(2) For any $i', j' \in P'$: if i' and j' are nonfaulty in Φ', then $B'_{i'}(\Phi'_{i'}) = B'_{j'}(\Phi'_{j'})$.

To prove these conditions, we use our observation that if Process i' is nonfaulty in Φ', then every process in P_i is nonfaulty in $\Lambda[\Phi']$. Hence, if all processes in P' are nonfaulty in Φ', then all processes in P are nonfaulty in $\Lambda[\Phi']$. Using condition 1 for the m-fault WBG algorithm B, we see that

$$\begin{aligned} B'_{i'}(\Phi'_{i'}) &= B_{i''}(\Lambda[\Phi']_{i''}) \\ &= \Lambda[\Phi'](0'') \\ &= \Phi'(0'), \end{aligned}$$

which proves condition 1 for B'.

Next, assume that the i' and j' are nonfaulty in Φ'. Since i'' and j'' are nonfaulty in $\Lambda[\Phi']$, condition 2 for B yields

$$\begin{aligned} B'_{i'}(\Phi'_{i'}) &= B_{i''}(\Lambda[\Phi']_{i''}) \\ &= B_{j''}(\Lambda[\Phi']_{j''}) \\ &= B'_{j'}(\Phi'_{j'}). \end{aligned}$$

This proves condition 2 for B'. We have thus constructed a 1-fault WBG algorithm for the three processes 0', 1', 2', contradicting Lemma 5. □

3. *Approximate and Infinite Solutions*

We now describe an approximate solution to the WBG problem that works in the presence of any number of faulty processes. By taking the limit of a sequence of such solutions, we obtain an exact solution using an infinite number of messages. In order to make the concept of an approximate solution meaningful, we assume that the set V of possible values is a set of real numbers.

For each integer $k > 0$, we define an algorithm $AG^{(k)}$ that requires k rounds of message passing. Rather than describing it in terms of formal scenarios, we will simply talk about processes sending messages to one another. Nonfaulty processes are constrained to follow the algorithm, while faulty ones may do anything. We assume that a faulty process sends every message that it is supposed to, although possibly with an incorrect value. However, every value it sends is assumed to be some element of V. It should be obvious how this description can be translated into a definition of mappings on i-scenarios.

Algorithm $AG^{(k)}$: The following k rounds of message passing are executed to compute the values $v_i^{(r)}$ for $i \in P$ and $1 \leq r \leq k$.

—Round 1:

* Process 0 sends the value v to every Process i.
* Each Process i sets $v_i^{(1)}$ equal to the value it receives from Process 0.

—Round r: $(1 < r \leq k)$

* Each Process j sends the value $v_j^{(r-1)}$ to every Process i.
* Each Process i sets $v_i^{(r)}$ equal to the maximum of the n values it receives.

Each Process i then sets v_i equal to the average of the k values $v_i^{(r)}$.

We now prove the following result about this algorithm, which shows that it is an approximate solution to the WBG problem.

THEOREM II. *If $|v| < D$ for all $v \in V$, then the algorithm $AG^{(k)}$ satisfies the following properties.*

(1) *If all processes are nonfaulty, then $v_i = v$ for every i.*
(2) *If Processes i and j are nonfaulty, then $|v_i - v_j| < 2D/k$.*

Note that no limit is placed upon the number of faulty processes. The proof of this theorem uses the following lemma.

LEMMA 6. *Assume that $|v| < D$ for all $v \in V$. If s_r, t_r are elements of V such that:*

$$s_r \leq t_{r+1}, \qquad t_r \leq s_{r+1}$$

for all r with $1 \leq r < k$, then

$$\left| \sum_{r=1}^{k} s_r - \sum_{r=1}^{k} t_r \right| < 2D.$$

PROOF. It follows from the first inequality of the hypothesis that

$$\sum_{r=1}^{k} s_r \leq s_k + \sum_{r=2}^{k} t_r.$$

From this, we deduce that

$$\sum_{r=1}^{k} s_r - \sum_{r=1}^{k} t_r \leq s_k - t_1 < 2D.$$

The symmetric argument, interchanging s and t, yields

$$\sum_{r=1}^{k} t_r - \sum_{r=1}^{k} s_r < 2D,$$

and combining the two inequalities proves the lemma. □

PROOF OF THEOREM. To prove the first property, we simply observe that if all processes are nonfaulty, then they correctly relay values, so all the $v_i^{(r)}$ equal v. To prove the second property, we note that if Processes i and j are nonfaulty, then they correctly relay the values of $v_i^{(r)}$ and $v_j^{(r)}$ to one another in round $r + 1$. It therefore follows that for each $r \geq 1$:

$$v_i^{(r)} \leq v_j^{(r+1)}, \qquad v_j^{(r)} \leq v_i^{(r+1)}.$$

The second property then follows immediately from the above lemma, substituting $v_i^{(r)}$ for s_r and $v_j^{(r)}$ for t_r. □

To construct an infinite-message solution to the WBG problem, we let each Process i take as its value of v_i the limit as k goes to infinity of the values obtained by the algorithms $\mathrm{AG}^{(k)}$. If the set V is unbounded, then this limit could be infinite, in which case some arbitrary preassigned value is used. This gives us the following.

Algorithm $AG^{(\infty)}$: Compute the values $v_i^{(r)}$ as described in Algorithm $\mathrm{AG}^{(k)}$, for all $i \in P$ and $r \geq 1$. For each i, define v_i to equal $sup\{v_i^{(r)} : r \geq 1\}$, where ∞ is interpreted to be some arbitrary fixed element of V.

We now show that $\mathrm{AG}^{(\infty)}$ is a "solution" to the WBG problem that can tolerate any number of faults. Of course, since it requires choosing a value based upon an infinite sequence of messages, it cannot be regarded as a solution in any practical sense.

THEOREM III. *If V is a bounded set of numbers, then $AG^{(\infty)}$ is an infinite m-fault WBG algorithm, for any m.*

PROOF. The proof is quite simple, and rests upon the observation that if i and j are nonfaulty, then

$$v_i^{(r)} \leq v_j^{(r+1)}, \qquad v_j^{(r)} \leq v_i^{(r+1)}$$

for all $r > 0$, which in turn implies that $sup\{v_i^{(r)}\} = sup\{v_j^{(r)}\}$. The details are left to the reader. □

Glossary of Notation

General Notation:

—n: the number of processes.
—P: the set $\{0, \ldots, n-1\}$ of processes.
—P^*: the set of finite sequences of processes.
—Π: the set of message paths from 0—sequences in P^* beginning with 0.
—Π_i: the set of message paths from 0 to i—sequences in Π ending in i.
—$\Pi^{(k)}$: set of message paths of length $\leq k$ in Π.
—$\Pi_i^{(k)}$: set of message paths of length $\leq k$ in Π_i.
—V: the set of all possible values v.

—scenario: a mapping Φ: $\Pi \rightarrow V$—specifying the value of the contents of every message.
—i-scenario: a mapping Φ_i: $\Pi_i \rightarrow V$—the part of a scenario "seen" by Process i.
—WBG algorithm B: a collection of mappings B_i from i-scenarios to V.

Notation for $n = 3$:

—$\delta(i, j)$: the signed, clockwise distance from i to j.
—$\sigma(\pi)$: the signed angular distance traveled by the path π.
—$\bar{r}$: r *mod* 3.
—$\Pi^{(r)}$: a scenario in which a faulty Process $\bar{r}$ relays each message π correctly unless $\sigma(\pi) = r$, in which case it relays the value F to $\overline{r-1}$ and the value T to $\overline{r+1}$.

Notation used in proof of Theorem I:

—P': the set of processes $\{0', 1', 2'\}$.
—λ: a mapping assigning to each process in P a process in P', which assigns at most m processes to each process in P'. Also, its extension to a mapping from message paths in P to message paths in P'.
—i'': an element in P that is assigned by λ to i', where $i = 0$, 1 or 2.
—Λ: a mapping that takes scenarios on P' into scenarios on P, defined by letting the value of $\Lambda[\Phi']$ on the message path π equal the value of Φ' on the path $\lambda(\pi)$.

ACKNOWLEDGMENT. We wish to thank Michael Fischer for discovering an error in an earlier version of this paper, and for his help in improving the presentation.

REFERENCES

1. LAMPORT, L., SHOSTAK, R., AND PEASE, M. The Byzantine Generals Problem *Trans Prog. Lang. Syst. 4*, 3 (July 1982), 382–401
2 PEASE, M., SHOSTAK, R , AND LAMPORT, L Reaching agreement in the presence of faults. *J. ACM 27*, 2 (Apr. 1980), 228–234.
3 GRAY, J. Notes on database operating systems. In *Operating Systems, an Advanced Course*, Lecture Notes in Computer Science 60, R Beyer, R M. Graham, and G. Seegmuller, Eds., Springer-Verlag, New York, 1978, pp. 393–481.

RECEIVED JANUARY 1981; REVISED MARCH 1982; ACCEPTED JUNE 1982