PROBABILISTIC LOWER BOUNDS FOR BYZANTINE AGREEMENT AND CLOCK SYNCHRONIZATION†

Anna Rochelle Karlin and Andrew Chi-Chih Yao
Computer Science Department
Stanford University
Stanford, California 94305

Abstract

We show that any probabilistic algorithm for reaching Byzantine agreement will fail with probability at least 1/3, if $t \geq \frac{n}{3} \geq 1$, where n is the total number of processors and t is the maximum number of possible faulty ones. We also prove that, for any n and t, any probabilistic algorithm that always terminates in less than $t+1$ rounds of communication will fail with probability greater than $\epsilon_{n,t}$, where $\epsilon_{n,t} > 0$ is a constant. For the clock synchronization problem of n processors with t possible faulty ones, we will prove that, if $t \geq \frac{n}{3} \geq 1$, the asymptotic proportion of synchronized time duration (over a long period of time) achievable by any probabilistic algorithm cannot exceed 2/3. These results generalize several well known impossibility results in fault-tolerant distributed systems for deterministic algorithms.

† Research supported by DARPA contract N00039-83-C-0136 and NSF grants MCS-82-03405 and MCS-83-08109.

1. Introduction

In a fault-tolerant distributed system, the generic problem of reaching certain kinds of agreement for independent processors is of central importance. Prominent among such problems are various versions of the Byzantine agreement problem and the clock synchronization problem. Several elegant impossibility results are known for these problems, which sharply characterize how much faultiness can be overcome . As these results apply to the worst case scenario for deterministic algorithms, questions have been raised [ ] as to whether they remain true when probabilistic algorithms with small failure probabilities are allowed. The purpose of this paper is to present extensions of a number of such impossibility results to the probabilistic case.

In the Byzantine agreement problem, a system of n processors wish to agree on a binary value v held by a certain processor, called the general. Up to t of the processors may be faulty. An algorithm for the problem will specify the rules governing the behavior of nonfaulty processors. In accordance with these rules the nonfaulty processors send and receive messages from each other in a pre-agreed number of synchronous rounds and decide on a value; the faulty processors may behave in an arbitrary manner. At the end of an execution of the algorithm, we say that Byzantine agreement has been achieved, when (1) all nonfaulty processors agree on the same value, and (2) if the general is nonfaulty, then all nonfaulty processors agree on his value. A fundamental question was, under what situations do there exist deterministic algorithms that always achieve Byzantine agreement.

There are two known basic results concerning this last question. Pease, Shostak, and Lamport [] showed that there can be no such algorithms when $t \geq n/3$, i.e., when at least $1/3$ of the processors are faulty. Fischer and Lynch [] showed that no such algorithm exists that terminates in less than $t+1$ rounds of communication. Both bounds are tight in that there are algorithms that terminate in $t+1$ rounds if $t \leq n/3$.

In this paper we will prove the following results for the Byzantine agreement problem: (A) For any probabilistic algorithm, if $t \geq n/3$, the faulty processors can behave in such a way that at least with probabilty $1/3$, the algorithm will fail to achieve Byzantine agreement; (B) For any probabilistic algorithm that terminates in less than $t+1$ rounds, the faulty processors can behave in such a way that the algorithm will fail to achieve Byzantine agreement with probability at least $\epsilon_{n,t}$, where $\epsilon_{n,t}$ are positive constants.

We pause to clarify what the "arbitrary behavior" of faulty processors means in the context of probabilistic algorithms. Informally, at each round, the faulty processors may

collude, examine all the messages they have received so far, and decide with what probability who should send what messages next. The procedure they use can be expressed as a probability distribution S over a suitable message space. The results (A), (B) above assert that there exists some S that will cause the algorithm to fail with at least a certain probability.

We now turn to the clock synchronization problem[ ], in which a system of n processors, each equipped with a clock, wish to synchronise their clocks by sending messages to each other at certain times. In [ ],[ ], [ ], algorithms were given that achieve synchronisation in reasonable models where t, the number of faulty processors, is less than $n/3$. It was conjectured by Lamport [ ] that no algorithm for synchonization is possible if $t \geq n/3$. This conjecture was proved by Dolev, Halpern and Strong [ ] in the model of [ ] for deterministic algorithms, in the worst case sense. It was posed as an open problem in [ ] to find a corresponding lower bound to the probabilistic case. In this paper, we prove that if $t \geq n/3$, for any probabilistic synchronization algorithm, the following is true:

(C) Over a long period of time, the expected fraction of the time at which all nonfaulty clocks are synchronized cannot exceed 2/3.

We remark that the above statement is of interest even for deterministic synchronization algorithms.

It is of interest to contrast our results with some of the recent work by other authors on probabilistic algorithms for Byzantine agreement. Rabin[ ] showed that Byzantine agreement can be achieved in constant expected rounds when processors are provided initially with some random bits externally generated. Recently, Bracha[ ] gave a probabilistic algorithm that reaches agreement in O(log(n)) expected rounds. This last result is not contradictory to our result (B) above, as a probabilistic algorithm may have a finite probability $\epsilon_{n,t}$ to last more than t rounds but $\epsilon_{n,t}$ can be small enough to maintain a much smaller expected number of rounds. We would also like to mention that for probabilistic algorithms that do not make errors, several authors (Bracha and Toueg[ ], Broder and Dolev[ ]) have observed that the impossibility results for deterministic algorithms still hold; our results are stronger than theirs.

2. Model and Results

2.1 Byzantine Agreement

We begin by formalizing the notion of a probabilistic algorithm for reaching Byzantine

agreement. We employ a modified version of the formalism given by Broder and Dolev[ ]. Without loss of generality, we let processor 0 be the general and denote by $m_0 \in \{0,1\}$ his initial value.

Each processor i is assumed to have access to an unbiased and independent coin C_i. The sequence of coin flips done by processor i to be used during round r is denoted $\overline{C}_{r,i}$.

For the protocols we consider, the behavior of processor i during round r can be described by a transition function $\delta_i(M_{r-1,i}, \overline{C}_{r,i})$ where $M_{r-1,i}$ is the set of messages received by processor i up till and including round $r-1$. The value of $\delta_i(M_{r-1,i}, \overline{C}_{r,i})$ is a vector $\overline{m}_{r,i} = (m^0_{r,i}, m^1_{r,i}, \ldots, m^n_{r,i})$, where $m^j_{r,i}$ is the message processor i sends processor j during round r. Also, we define a binary decision function $\mu_i(M_{r,i}, \overline{C}_{r,i})$ which is the probability that the value v_i that processor i decides upon is 1 if the protocol terminates after r rounds and $M_{r,i}$ is the set of messages received. The complete specification of a protocol $\mathcal{A}$ is a transition function δ_i and a decision function μ_i for every processor i. Let $\mathcal{L}_n$ denote the set of all protocols $\mathcal{A}$.

A *scenario* σ consists of a value for m_0, a set F of faulty processors and a specification of arbitrary transition functions δ'_f for each $f \in F$ which dictate the behavior of the faulty processors. The transition functions δ'_f are generally different from δ_f and may be chosen so as to minimize the success of the algorithm.

Consider an r-round agreement protocol $\mathcal{A}$ and any scenario σ in which $|F| \leq t$. Given n random strings $\rho_1, \rho_2, \ldots, \rho_n$ $(\rho_i \in \{0,1\}^\infty)$, an execution of protocol $\mathcal{A}$ is completely determined by interpreting ρ_i as the coin tosses used by processor i. We call such an execution a *run of* $\mathcal{A}$, and we say it is successful if the following two conditions hold after r rounds:

(1) *Validity*: If processor $i \notin T$, then $v_i = m_0$.

(2) *Consistency*: $v_i = v_j$ for all processors $i, j \notin T$.

Let $e(\mathcal{A}, \sigma)$ be the probability that a random run of $\mathcal{A}$ using σ will not be successful, and let $e(\mathcal{A}) = \sup\{e(\mathcal{A}, \sigma) | \text{ all } \sigma\}$. Thus, $e(\mathcal{A})$ is the maximum error probability that the faulty processors can force $\mathcal{A}$ to suffer.

We prove the following theorems.

Theorem 1: If $3 \leq n \leq 3t$, then $e(\mathcal{A}) \geq 1/3$ for all $\mathcal{A} \in \mathcal{L}_n$.

Theorem 2: If $\mathcal{A} \in \mathcal{L}_n$ is a $k+1$ round protocol where $1 \leq k < t$ and $n \geq 2t+2$, then $e(\mathcal{A}) \geq \frac{1}{2}(\frac{t}{n(k+1)})^k$.

he processors in F will employ in sending messages. Let $Y_{n,t}$ be the set of all scenarios.

Given any $\mathcal{B} \in \mathcal{F}_n$, $\sigma \in Y_{n,t}$, the stochastic behavior of the system of processors is now well defined. Let $b > 0$ be any number. At any Newtonian time $t > 0$, we observe if the system is *b-synchronized*, i.e. $|C_i(t) - C_j(t)| < b$ for all $i, j \notin F$. For any $T > 0$, let $s_1'(\mathcal{B}, \sigma, T)$ be the probability that the system is b-synchronized at a random Newtonian time $t \in [0, T]$. Let $s_b''(\mathcal{B}, \sigma) = \lim \sup\{s_1'(\mathcal{B}, \sigma, T) | 0 < T < \infty\}$, which measures the probability of b-synchronization asymptotically. Finally, define $s_{b,n,t}(\mathcal{B}) = \inf\{s_b''(\mathcal{B}, \sigma) | \sigma \in Y_{n,t}\}$, the success probability of algorithm $\mathcal{B}$ for the worst possible adversary.

We will prove the following result.

Theorem 3: If $t \geq n/3 \geq 1$, then $s_{b,n,t}(\mathcal{B}) \leq 2/3$ for all $\mathcal{B} \in \mathcal{F}_n$.

3. Proof of Theorem 1

For some scenario σ, let $M_{r,i}^j$ for $0 \leq j \leq N$ be an enumeration of the possible values of $M_{r,i}$. Define a *view* V_i to be a probability distribution over the set $M_{r,i}^j$ with respect to σ. In other words, $V_i(M_{r,i}^j)$ is the probability that $M_{r,i}^j$ was the set of messages received by processor i in r rounds under σ.

Theorem 1: If $3 \leq n \leq 3t$, then any protocol A has $e(A) > \frac{1}{3}$.

Proof: Since $n \leq 3t$, the processors can be partitioned into 3 nonempty sets A, B and D, each with cardinality at most t. We shall construct three scenarios α, β and σ such that in α, $F = B$, in β, $F = A$ and in σ, $F = D$. The general (processor 0) will be one of the processors in D. In α, $m_0^\alpha = 0$ and in β, $m_0^\beta = 1$. Let D={0,..,n/3-1}, A={n/3,..,2n/3-1} and B={2n/3,..,n-1}.

The behavior of the faulty processors is described as follows:

(1) For $i \in D$, if $\delta_i^\alpha(M_{r-1,i}, \overline{C}_{r,i}) = \overline{m}_{r,i}$ and $\delta_i^\beta(Q_{r-1,i}, \overline{C}_{r,i}) = \overline{q}_{r,i}$, then in round r under scenario σ, processor i sends $(m_{r,i}^0, \ldots, m_{r,i}^{2n/3-1}, q_{r,i}^{2n/3}, \ldots, q_{r,i}^{n-1})$.

(2) For $i \in A$, $\delta_i^\alpha := \delta_i^\sigma$.

(3) For $i \in B$, $\delta_i^\beta := \delta_i^\sigma$.

By induction on the number of rounds of communication, one may verify that

(1) $V_i^\alpha = V_i^\sigma$ for $i \in A$ and

(2) $V_i^\beta = V_i^\sigma$ for $i \in B$.

Since in scenario α, $m_0^\alpha = 0$, $\text{Prob}(v_i^\alpha = 0) \geq 1 - e(A)$ for $i \in A$ by validity. Therefore $\text{Prob}(v_i^\sigma = 0) \geq 1 - e(A)$. Similarly, $\text{Prob}(v_i^\sigma = 1) \geq 1 - e(A)$ for $i \in B$.

Thus $Prob(v_i^\sigma = v_j^\sigma \text{ for } i \in A \text{ and } j \in B) \leq 2e(\mathcal{A})$. By consistency for σ, we must have $2e(\mathcal{A}) \geq 1 - e(\mathcal{A})$ and so $e(\mathcal{A}) \geq \frac{1}{3}$. □

4. Proof of Theorem 2

We now give a lower bound on the error probability for a k round algorithm. The idea of the proof is to consider a sequence of scenarios, such that each adjacent pair of scenarios has some processor whose message receipt distribution is the same. If in the first and last scenarios the general is nonfaulty with initial values 0 and 1 respectively, then the difference between the expected decision of a nonfaulty processor in the first scenario and another nonfaulty processor in the last scenario must be large. By summing over differences in expected decisions in adjacent scenarios and comparing that sum to the total difference just described, we obtain a lower bound on the error. The sharpest bound obtained by this method requires constructing a minimal length sequence of scenarios.

We say that two views V_i and V_l are consistent if there exists a scenario σ where processors i and l are nonfaulty and i's view is V_i and j's view is V_j . If processor i has the view V_i, then the expected value of v_i is denoted $g_i(V_i)$ and is computed by the formula

$$g_i(V_i) = \sum_j \mu_i(M^j_{r,i}, \overline{C}_{r,i}) V_i(M^j_{r,i}).$$

Lemma 1: Let $\mathcal{A}$ be any protocol in $\mathcal{L}_n$. If V_i and V_l are consistent, then

$$|g_i(V_i) - g_l(V_l)| \leq e(\mathcal{A}).$$

Proof: This follows immediately from the fact that

$$|g_i(V_i) - g_l(V_l)| \leq Prob(v_i \neq v_l) \leq e(\mathcal{A})$$

by consistency. □

We now define a sequence of scenarios $S_0, S_1, \ldots, S_M$. Each S_i, $1 \leq i \leq M$, can be specified by a $k+1$-tuple $(i_1, i_2, \ldots, i_k, x)$ where $0 \leq i_j < \lceil \frac{nk}{t} \rceil$ and $x \in \{1, 2\}$. The S_i's, $1 \leq i \leq M$, are ordered lexicographically by the tuples (starting from $(0, 0, \ldots, 0, 1, 1)$), so that $M = 2(\lceil \frac{nk}{t} \rceil)^k - 2$. We also associate with each scenario S_i that has $x = 1$, a number p_i such that if $S_i = (i_1, i_2, \ldots, i_k, 1)$ and $S_{i+2} = (i'_1, i'_2, \ldots, i'_k, 1)$ then $p_i \neq i_j$ and $p_i \neq i'_j$ for $1 \leq j \leq k$. We also require that $p_i \neq p_{i+2}$ for any i. We know that such p_i's exist for $n \geq 2t + 2$ and $k \leq t$. (Let $p_0 = \lfloor \frac{nk}{t} \rfloor$).

The complete description of scenario $C_j = (i_1, i_2, \ldots, i_k, x)$ for $0 < j \leq M$ is as follows:

(1) Processors $\{0, i_j \cdot \lfloor \frac{t}{k} \rfloor + l \mid 1 \leq j \leq k, 0 \leq l < \lfloor \frac{t}{k} \rfloor\} \in F$.

(2) This scenario consists of $k+1$ rounds.

(3) In round 0, processor 0 takes $\delta_0(1, \overline{C}^1_{0,0}) = \overline{m}_{0,0}$ and $\delta_0(0, \overline{C}^2_{0,0}) = \overline{q}_{0,0}$ and sends the vector $(m^0_{0,0}, m^1_{0,0}, \ldots, m^{(i_1+1)\lfloor \frac{t}{k} \rfloor - 1}_{0,0}, q^{(i_1+1)\lfloor \frac{t}{k} \rfloor}_{0,0}, \ldots, q^{n-1}_{0,0})$. All other faulty processors behave as if they were nonfaulty.

(4) In round j $(1 \leq j < k)$,processor $i_j + l$, for $0 \leq l < \lfloor \frac{t}{k} \rfloor$ takes $\delta_{i_j+l}(M_{i_j+l,j-1}, \overline{C}^1_{j,i_j+l}) = \overline{m}_{j,i_j+l}$ and $\delta_{i_j+l}(Q_{i_j+l,j-1}, \overline{C}^2_{j,i_j+l}) = \overline{q}_{j,i_j+l}$ and sends the vector $(m^0_{j,i_j+l}, m^1_{j,i_j+l}, \ldots, m^{(i_{j+1}+1)\lfloor \frac{t}{k} \rfloor - 1}_{j,i_j+l}, q^{(i_{j+1}+1)\lfloor \frac{t}{k} \rfloor}_{j,i_j+l}, \ldots, q^{n-1}_{j,i_j+l})$. All other faulty processors behave as if they were nonfaulty. $M_{i_j+l,j-1}$, $0 \leq l < \lfloor \frac{t}{k} \rfloor$, is the set of messages actually received by processor $i_j + l$.

$Q_{i_j+l,j-1}$, $0 \leq l < \lfloor \frac{t}{k} \rfloor$, is the set of messages actually received except that the following messages are used instead of the corresponding messages actually received:

$$q^{i_j+l}_{0,0}, q^{i_j+l}_{1,i_1+y}, \ldots, q^{i_j+l}_{j-1,i_j+y}; \quad 0 \leq y < \left\lfloor \frac{t}{k} \right\rfloor.$$

If $i_j + l = 0$ for some i_j and l, then in $Q_{0,j-1}$ use 0 as the starting value.

(5) In round k ,processor $i_k + l$, for $0 \leq l < \lfloor \frac{t}{k} \rfloor$, takes $\delta_{i_k+l}(M_{i_k+l,k-1}, \overline{C}^1_{k,i_k+l}) = \overline{m}_{k,i_k+l}$ and $\delta_{i_k+l}(Q_{i_k+l,k-1}, \overline{C}^2_{k,i_k+l}) = \overline{q}_{k,i_k+l}$ and actually sends a vector in accordance with one of the following cases:

(a) If $x = 1$ and $p_{j-2} > p_j$, the messages sent are

$$(m^0_{k,i_k+l}, m^1_{k,i_k+l}, \ldots, m^{p_j(\lfloor \frac{t}{k} \rfloor)}_{k,i_k+l}, q^{p_j(\lfloor \frac{t}{k} \rfloor)+1}_{k,i_k+l}, \ldots, q^{n-1}_{k,i_k+l}).$$

(b) If $x = 1$ and $p_{j-2} < p_j$, the messages sent are

$$(q^0_{k,i_k+l}, q^1_{k,i_k+l}, \ldots, q^{p_j(\lfloor \frac{t}{k} \rfloor)-1}_{k,i_k+l}, m^{p_j(\lfloor \frac{t}{k} \rfloor)}_{k,i_k+l}, \ldots, m^{n-1}_{k,i_k+l}).$$

(c) If $x = 2$, the messages sent are $(m^0_{k,i_k}, \ldots, m^{n-1}_{k,i_k})$.

All other faulty processors behave as if they were nonfaulty. As before, $M_{i_k+l,k-1}$, $0 \leq l < \lfloor \frac{t}{k} \rfloor$, is the set of messages actually received. Similarly, $Q_{i_k+l,k-1}$, $0 \leq l < \lfloor \frac{t}{k} \rfloor$, is the set of messages actually received except that the following messages are used instead of the corresponding messages actually received:

$$q^{i_k+l}_{0,0}, q^{i_k+l}_{1,i_1+y}, \ldots, q^{i_k+l}_{k-1,i_k+y}; \quad 0 \leq y < \left\lfloor \frac{t}{k} \right\rfloor.$$

If $i_k + l = 0$ for some i_k and l, then in $Q_{0,k-1}$ use 0 as the starting value.

In S_0, all processors behave as if they are nonfaulty and the general's initial value is 0, and in S_M, the general can be considered nonfaulty, with initial value 1.

Lemma 2: For each l, $0 \leq l < M$, there is an i, $0 \leq i \leq n-1$ and a view V_i such that V_i appears in both S_l and S_{l+1}.

Proof: Using the scenarios just constructed, we can verify that for S_j with $x = 1$, $V_{p_{j-1}}$ appears in both S_j and S_{j+1} and for S_j with $x = 2$, $V_{p_{j-1}}$ appears in both S_j and S_{j+1}. □

We now have all we need to prove the main theorem.

Theorem 2: If $A \in \mathcal{L}_n$ is a $(k+1)$-round protocol where $1 \leq k < t$ and $n \geq 2t+2$, then $\epsilon(A) > \frac{1}{2}(\frac{t}{n(k+1)})^k$.

Proof: By lemma 2, we can let $V_{i_0}, V_{i_1}, .., V_{i_M}$ be a sequence of views such that V_{i_l} appears in both S_l and S_{l+1} for $0 \leq l < M$. We know that $|g_{i_l}(V_{i_l}) - g_{i_{l+1}}(V_{i_{l+1}})| \leq \epsilon(A)$ by lemma 1.

Thus $|g_{i_0}(V_{i_0}) - g_{i_M}(V_{i_M})| \leq \sum_{0 \leq l < M} |g_{i_l}(V_{i_l}) - g_{i_{l+1}}(V_{i_{l+1}})| \leq M\epsilon(A)$. But $g_{i_0}(V_{i_0}) \geq 1 - \epsilon(A)$ and $g_{i_M}(V_{i_M}) \leq \epsilon(A)$ by validity. Thus $M\epsilon(A) \geq 1 - 2\epsilon(A)$ and so $\epsilon(A) \geq \frac{1}{M+2}$. Substituting for M, we get $\epsilon(A) \geq \frac{1}{2}(\frac{t}{n(k+1)})^k$. □

We may completely bypass the issue of extending this result to authenticated algorithms by showing that the proof goes through when the faulty processors are only allowed not to transmit information. The corresponding deterministic result was shown in Dolev and Strong[ ].

Corrolary:

If A is any $(k+1)$-round agreement protocol *with authentication* where $1 \leq k < t$ and $n \geq 2t+2$, then $\epsilon(A) > \frac{1}{2}(\frac{t}{n(k+1)})^k$.

Proof Sketch: In the scenarios defined, let the faulty processors send no messages to those processors they previously sent faulty information to. The proof goes through almost analogously. □

References

Bracha, G. [1984] "A randomized Byzantine agreement with an O(log(n)) expected rounds," manuscript.

Broder, A. and D. Dolev [1984] "Flipping coins in many pockets (Byzantine agreement on uniformly random values)," *Proc. of the 25th Annual Symposium on Foundations of Computer Science.*

Dolev, D. and R. Strong [1983] "Authenticated Algorithms for Byzantine Agreement," *SIAM Journal of Computing*, Vol. 12, 656-666.

Fischer, M. and N. Lynch [1982] "A lower bound for the time to assure interactive consistency," *Information Processing Letters*, Vol. 14, 183-186.

Pease M., R. Shostak and L. Lamport [1980] "Reaching agreement in the presence of faults," *Journal of the ACM*, Vol. 27, 228-234.

Rabin, M. [1983] "Randomized Byzantine generals," *Proc. of the 24th Annual Symposium on Foundations of Computer Science.*