Ben-Or asynchronous Byzantine agreement algorithm

Introduction

The idea of the Byzantine agreement problem is to have a system of N processes, t of which are faulty. Each correct process starts with either 0 or 1, and after running the algorithm, all correct processes must end up with either 0 or 1 based on the following rules:

- All correct processes end up with the same value.
- If all correct processes start with the same value v, then all correct processes must end with v.
- All correct processes must terminate (with probability 1).

Ben-Or algorithm allows the network to be asynchronous and does not restrict the malevolence of faulty processes. It works under assumption $t < \frac{N}{5}$.

Protocol

Let N - number of all processes, t - number of faulty processes, $t < \frac{N}{5}$
Let's bring up the pseudo code[1]:

Process P: Initial value x_p.
step 1: Set $r \coloneqq 1$.
step 0: Send the message (1, r, x_p) to all the processes.
step 2: Wait till messages of type (1, r, *) are received from $N - t$ processes. If more than $(N - t)/2$ messages have the same value v, then send the message (2, r, v, D) to all processes. Else send the message (2, r, ?) to all processes.
step 3: Wait till messages of type (2, r, *) arrive from $N - t$ processes.
(a) If there are at least $t + 1$ D-messages (2, r, v, D), then set $x_p \coloneqq v$.
(b) If there are more than $(N + t)/2$ D-messages then **decide** v.
(c) Else set $x_p = 1$ or 0 each with probability $\frac{1}{2}$.
step 4: Set $r \coloneqq r + 1$ and go to step 1.

Processes locally keep track of current round number, and store current binary value.
Within each round they go through 2 stages, messages from each round from now on will be called type 1 or 2 messages.

Correctness

Let's start with proving the following lemma:

Lemma 1. *If all correct processes start with the value v, then within one round they will all decide v.*

Proof. Each process broadcasts message $(1,\ 1,\ v)$.
Process then waits for $N - t$ messages.
At most t messages came from faulty processes, which means that at least $N - 2 \cdot t$ messages are correct. Because $N - 2 \cdot t > \frac{N+t}{2}$, all the correct processes send message $(2,\ 1,\ v,\ D)$.
Among $N - t$ accepted type 2 messages at most t are incorrect, which means that step 3(b) will not execute.
Again, because at least $N - 2 \cdot t$ of type 2 messages are correct, then more than $\frac{N+t}{2}$ of them will have the same value (v). That means that every process will **decide** v this round. □

Lemma 2. *If process sets v in step 3(a) in round r, then it can't set $\neg v$ in the same round.*

Proof. Let's say that process saw $\geq t+1$ messages $(2,\ r,\ 0,\ D)$ and $\geq t+1$ messages $(2,\ r,\ 1,\ D)$. Let A_x be equal to the number of messages $(2,\ r,\ x,\ D)$ originated from correct processes with $x = 0, 1$. Of course $A_0 + A_1 \geq t + 2$.
Every process responsible for A_x saw more than $\frac{N-t}{2}$ messages $(1,\ r,\ x)$ from correct process. But that means that there are more than $N - t$ correct processes, which is a contradiction. □

Then let's look at the next lemma, which states that the processes will decide with at most 2 round window.

Lemma 3. *If for some round r, some correct process decides v in step 3(b), then all other correct processes will decide v within the next round.*

Proof. Let P be the process that has decided on v. In order to **decide** v, P must have received more than $\frac{N+t}{2}$ messages $(2,\ r,\ \_,\ D)$. That means that more than $\frac{N-t}{2}$ correct processes sent that message. Let A_x be the number of $(2,\ r,\ x,\ D)$ messages from correct processes for $x = 0, 1$.
Because P has decided v, it follows that $A_{\neg v} \leq t$. On top of that $A_v + A_{\neg v} > \frac{N-t}{2}$.
Simple calculations show that $A_v \geq t+1$. It means that every other process will set v in step 3(a). From lemma 2 we know that it will be the only value that they will set.
It follows that in the next round every correct process will start with v, so lemma 1 can be applied. □

Time complexity

In each round processes have probability of at least $2^{-(N-t)+1}$ of all setting the same value, which using *Lemma 1* means that the run algorithm would end next round. That means that expected number of rounds is bounded by $O(2^{N-t})$.
The following theorem shows tighter bound under stricter assumptions:

Theorem 1. *[1] If $t = O(\sqrt{N})$ then the expected number of rounds to reach agreement in this protocol is constant, i.e., it does not depend on N.*

References

[1] Michael Ben-Or. Another Advantage of Free Choice: Completely Asynchronous Agreement Protocols. In *Proceedings of the second annual ACM symposium on Principles of distributed computing*, pages 27-30. ACM, 1983, doi: 10.1145/800221.806707.