Assignment 7

Problem Statement Give an algorithm for solving ϵ-approximate agreement with any $0 < \epsilon < 1$ for inputs in $[0, 1]$ that has $O\left(\log_2\left(\frac{1}{\epsilon}\right)\right)$ step complexity. Prove your algorithm is correct.

Algorithm. *(This algorithm is inspired by the one presented in-class as well as the one shown in the Attiya textbook, page 352.)* Let the processes be denoted $p_0, ..., p_{n-1}$ where each process p_i receives input $x_i \in [0, 1]$. The processes share $r = \lceil \log_2(1/\epsilon) \rceil + 1$ atomic snapshot objects denoted $S_0, ..., S_{r-1}$. These objects support: (1) UPDATE(S, i, v) where S refers to the object, i is the index to update, and v is updated value and (2) SCAN(S). Initially all entries of S_i are set to $\perp$.

The high-level description of the algorithm is as follows: each process begins with their input x_i as their desired output. For iteration k where k ranges from 0 to $r - 1$, each process write their desired output into the atomic snapshot object S_k and scans S_k. They change their desired output to the mid-point between the largest and smallest values seen in S_k. After r iterations, each process outputs their desired value. The associated pseudo-code is shown in Algorithm 1.

Each process takes a constant number of steps for each iteration of the for-loop. Since the for-loop runs for $\lceil \log_2(1/\epsilon) \rceil + 1 \in O(\log_2(1/\epsilon))$ iterations, the algorithm has the required step complexity.

Algorithm 1 Approximate agreement with $O\left(\log_2\left(\frac{1}{\epsilon}\right)\right)$ step complexity: code for process p_i.

```
r ← ⌈log_2(1/ε)⌉ + 1
v_{i,0} ← x_i
for k from 0 to r − 1 do
    UPDATE(S_k, i, v_{i,k})
    V_{i,k} ← SCAN(S_k)
    m_{i,k} ← min{x ∈ V_{i,k} : x ≠ ⊥}
    M_{i,k} ← max{x ∈ V_{i,k} : x ≠ ⊥}
    v_{i,k+1} ← (m_{i,k} + M_{i,k}) / 2
end for
output v_{i,r}
```

Proof of Correctness. The termination condition is easy to see, every process executes a for-loop for a finite number of iterations and every iteration is wait-free. Validity holds by Lemma 1. ϵ-agreement holds by Lemma 2.

Lemma 1. *Algorithm 1 satisfies the validity condition.*

Proof. We will show that the value $v_{i,k}$ of each process p_i satisfies the validity condition at iteration k of the for-loop by induction on the number of iterations. Initially $v_{i,0} = x_i$ for every p_i so the base case holds. Suppose the validity condition holds for iterations $0, 1, ..., k$ on all processes. We show that the condition must hold for iteration $k + 1$. During iteration $k + 1$, the process updates S_{k+1} with $v_{i,k+1}$, scans S_{k+1}, and finds the minimum and maximum elements ($m_{i,k+1}$ and $M_{i,k+1}$ respectively) in $V_{i,k+1}$. By the induction hypothesis all entries of $V_{i,k+1}$ satisfies the validity condition. Thus there exists inputs x_1, x_2, x_3, x_4 such that $x_1 \leq m_{i,k+1} \leq x_2$ and $x_3 \leq M_{i,k+1} \leq x_4$. Then the desired value for iteration $r + 1$ satisfies:

$$\min\{x_1, x_3\} \leq \frac{m_{i,k+1} + M_{i,k+1}}{2} \leq \max\{x_2, x_4\}.$$

Since $\min\{x_1, x_3\}$ and $\max\{x_2, x_4\}$ are inputs of some process, $v_{i,k+2}$ at iteration $k+1$ satisfies the validity condition for every process p_i. $\square$

Lemma 2. *Algorithm 1 satisfies the ϵ-agreement condition.*

Proof. Let $v_{i,k}$ and $v_{j,k}$ be the desired values of process p_i and p_j in iteration k. We show that $|v_j - v_i| \leq \frac{1}{2^k}$ by induction on the number of iterations. The base case is true since the inputs are taken from the interval $[0, 1]$. Suppose the claim is true for iterations $0, 1, ..., k$. Show that the claim holds on iteration $k+1$. Consider process p_i in iteration $k+1$ of the for-loop. p_i writes the current desired value $v_{i,k+1}$ into S_{k+1} and scans S_{k+1} into $V_{i,k+1}$. Let any $v_{j,k+1} \in V_{i,k+1}$ such that $j \neq i$ and $v_{j,k+1} \neq \perp$. Consider the views $V_{i,k}$ and $V_{j,k}$ from iteration k. Note that $v_{i,k+1} = \frac{m_{i,k}+M_{i,k}}{2}$ and $v_{j,k+1} = \frac{m_{j,k}+M_{j,k+1}}{2}$ where $m_{i,k}$, $M_{i,k}$, $m_{j,k}$, $M_{j,k}$ are the largest and smallest elements in $V_{i,k}$ and $V_{j,k}$ respectively. Since atomic snapshot objects are linearizable, either p_j or p_i performed the SCAN operation first. Assume the former. By the induction hypothesis, $|m_{i,k} - M_{i,k}| \leq \frac{1}{2^k}$, $|m_{j,k} - M_{j,k}| \leq \frac{1}{2^k}$, $|m_{i,k} - M_{j,k}| \leq \frac{1}{2^k}$, and $|m_{j,k} - M_{i,k}| \leq \frac{1}{2^k}$. Since p_i performed SCAN after p_j, $V_{i,k}$ contains values $m_{j,k}$, $M_{j,k}$, and $v_{j,k}$. Either $\frac{1}{2^{k+1}} \geq |v_{i,k+1} - m_{i,k}| \geq |v_{i,k+1} - m_{j,k}| \geq |v_{i,k+1} - v_{j,k+1}|$ or $\frac{1}{2^{k+1}} \geq |v_{i,k+1} - M_{i,k}| \geq |v_{i,k+1} - M_{j,k}| \geq |v_{i,k+1} - v_{j,k+1}|$. See Figure 1.

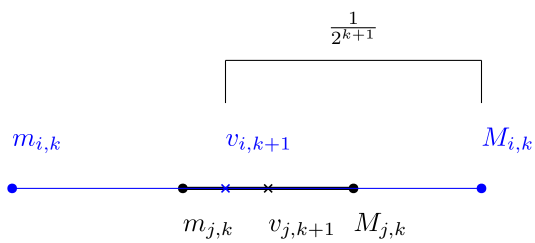

Figure 1: Situation where $\frac{1}{2^{k+1}} \geq |v_{i,k+1} - M_{i,k}| \geq |v_{i,k+1} - M_{j,k}| \geq |v_{i,k+1} - v_{j,k+1}|$.

The same is true if p_j performed SCAN after p_i as p_j will set $v_{j,k+1}$ such that $|v_{i,k+1} - v_{j,k+1}| \leq \frac{1}{2^{k+1}}$. Thus $|v_{j,k} - v_{i,k}| \leq \frac{1}{2^k}$ at iteration k for all $k \in \mathbb{N}$. Since $\frac{1}{2^r} \leq \epsilon$ where r is the number of iterations, ϵ-agreement condition is satisfied after r iterations of the for-loop. $\square$

Extra credit: extend the algorithm to run in

$$O\left(\log_2\left(\frac{\max\{|x_0|, ..., |x_{n-1}|\}}{\epsilon}\right)\right).$$

steps when the inputs $x_0, ..., x_{n-1}$ are arbitrary real numbers.

Algorithm. *(Referenced the Attiya textbook page 355.)* As before we have n processes $p_0, ..., p_{n-1}$ where the input to p_i is x_i. We use one atomic snapshot objects S where each entry $S[i]$ is $\langle x_i, c_i, v_i\rangle$ which are the input, current round counter, and value of process p_i respectively. The operations on S are the same as before. Initially all values in S are set to $\perp$. Further define the modified logarithm function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ as

$$f(x) = \begin{cases} 0 & \text{if } x = 0 \\ \log\left(\frac{x}{\epsilon}\right) & \text{otherwise} \end{cases}$$

The general idea for this algorithm is the similar to the one for the previous problem. At each round a process p_i reads the values in the atomic snapshot object S and updates its local value to be the midpoint of the range of the observed values. However, since processes are initially unaware of the range of the inputs, p_i must keep a round counter c_i in $S[i]$ to let all the other processes know which step it is on. At every iteration, p_i will look at a process p_j with the largest counter c_j and jump forward to round c_j by taking the mid-point of values from all processes at round c_j. Further, p_i will keep an estimate of the maximum number of rounds, $r_{\max}$, using all the values of x_j it has seen and exit the loop when c_i reaches $r_{\max}$. See Algorithm 2 for the associated pseudo-code.

Every iteration of the while-loop takes a constant number of steps. Since the while-loop terminates after $\lceil\log\left(2 \cdot \frac{\max\{|x_0|,\ldots,|x_{n-1}|\}}{\epsilon}\right)\rceil + 1$ iterations by Lemma 3, the step complexity is as required.

Algorithm 2 Approximate agreement with $O\left(\log_2\left(\frac{\max\{|x_0|,\ldots,|x_{n-1}|\}}{\epsilon}\right)\right)$ step complexity: code for process p_i.

$v_i \leftarrow x_i$
$c_i \leftarrow 1$
$r_{\max} \leftarrow 2$
while $r_{\max} > c_i$ **do**
    $\textsc{update}(S, i, \langle x_i, c_i, v_i\rangle)$
    $V \leftarrow \textsc{scan}(S)$
    $m_x \leftarrow \min\{x_k : \langle x_k, c_k, v_k\rangle \in V \text{ and } x_k \neq \bot\}$
    $M_x \leftarrow \max\{x_k : \langle x_k, c_k, v_k\rangle \in V \text{ and } x_k \neq \bot\}$
    $r_{\max} \leftarrow f(M_x - n_x)$
    $c_i \leftarrow \max\{c_k : \langle x_k, c_k, v_k\rangle \in V \text{ and } c_k \neq \bot\}$
    $m_v \leftarrow \min\{v_k : \langle x_k, c_k, v_k\rangle \in V \text{ and } c_k = c_i\}$
    $M_v \leftarrow \max\{v_k : \langle x_k, c_k, v_k\rangle \in V \text{ and } c_k = c_i\}$
    $v_i \leftarrow \frac{m_v + M_v}{2}$
    $c_i \leftarrow c_i + 1$
end while
output v_i

Proof of Correctness The termination condition holds by Lemma 3. The validity condition holds by an argument similar to Lemma 1 and the ϵ-agreement condition holds by an argument similar to Lemma 2.

Lemma 3. *For every process p_i, Algorithm 2 terminates after c iterations where*

$$c = \left\lceil \log\left(2 \cdot \frac{\max\{|x_0|, \ldots, |x_{n-1}|\}}{\epsilon}\right)\right\rceil + 1.$$

Proof. Suppose that there exists a process p_i whose while-loop iterates more than c times. The exist condition is $c_i \geq r_{\max}$. Since c_i is initially 1 and increments by at least 1 in every iteration, $r_{\max}$ must be greater than c. However $r_{\max} = f(M_x - n_x)$ where M_x is the largest input that p_i has seen and m_x is the smallest. Observe that $M_x - m_x \leq 2 \cdot \max\{x_0, \ldots, x_{n-1}\}$. Thus $f(M_x - m_x) \leq c$. This contradicts our assumption so the while-loop of every process terminates in at most c steps. □