**Algorithm 4.2: Host application to receive and send metric information**

**Input:** packets from the controller: Ci, packet from nearby host: Pi, number of packets to transfer: N

**Output:** packet to another host: Po, packet to Controller Co

1: Pi ← Server listening to another host (separate thread)

2: **While** true

3: Ci ← Server listening to Controller

4: A ← get the list of nearby hosts from Ci

5: W ← wait group

6: **For** Hs ← A (Iterate through every host IP address present in A)

7: Add 1 to W

8: Send N packets to Hs containing the current timestamp. After sending N packets remove 1 from W. (separate thread)

9: **End For**

10: Wait until W becomes 0 (Stops from listening to controller until all packets are sent from this host)

11: **End While**

12: Initialize Mp: Map hosts to array of delay

13: **Process Packet (Pi)**

14: Ts, Hs ← Pi

15: If first packet in the sequence start timer of 2s for Hs (Separate thread)

16: D ← Tcurrent – Ts

17: Add D to Mp[Hs]

18: If all N packets received then Send Packet Out (Hs) (Contains delay and jitter calculated from N packets)

19: Else If the timer is finished Send Packet Out (Hs) (Contains delay and jitter calculated from received packets)

20: **Finish Process Packet**

**Algorithm 5.1: Predicting future metrics using GRU model**

**Input:** GRU Model: M, Metric obtained at ith time sequence (5 second Interval): Li, Number of Sequence to consider: Ns, Number of Links: Ni, Number of metric parameters Np

**Output:** Predicted Metrics Pi

1: Li ← Timer Function called every 5 seconds returns the current capture of metrics (The normalized values V­ij ∀ (i ∈ [1, Ni], j ∈ [1, Nj]) are represented as matrix of size (Ni,Np) )

2: Initialize Linked List B

3: **While** True

4: **If** length(B) < Ns **then**

5: Push Back (B, Li)

6: **else**

7: Pop Front (Li)

8: F ← Convert B into 3-dimensional array of size (Ni, Ns, Np)

9: Pi ← M(F)

10: **endif**

11: **return** Pi