A Fault-Tolerant Clock Synchronization and Geometry Determination Protocol

Mahyar Malekpour
NASA Langley Research Center
AIAA SciTech 2018, 11 January 2018
Kissimmee, Florida

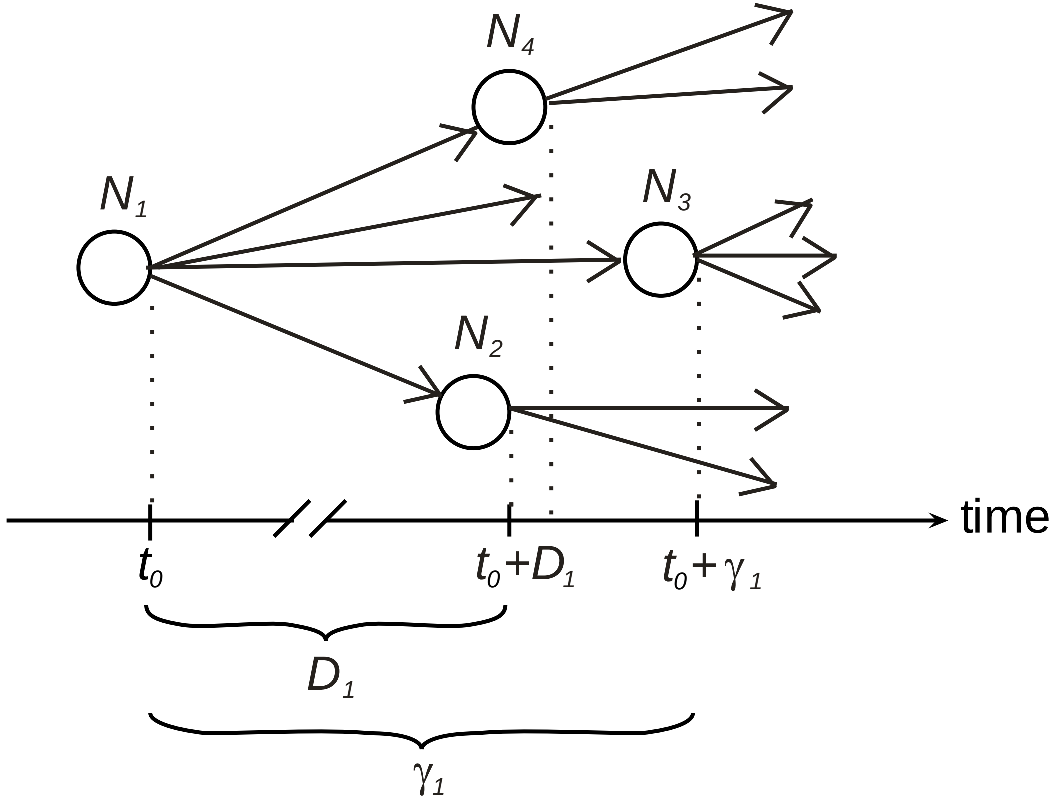
Communication And Synchronization

- Distributed systems are integral part of safety-critical computing applications, necessitating system designs that incorporate complex fault-tolerant resource management functions to provide globally coordinated operations with ultra-reliability
- Distributed systems are modeled as graphs, nodes and edges, with wired/wireless communication links
- Robust clock synchronization is a required fundamental service
- Faults add complexity, various types from benign to arbitrary (Byzantine)

What Is Synchronization?

- Local oscillators/hardware clocks operate at slightly different rates, thus, they drift apart over time
- Local logical clocks, i.e., timers/counters, may start at different initial values
- The synchronization problem is to adjust the values of the local logical clocks so that nodes achieve synchrony and remain synchronized despite the drift of their local oscillators
- Application – Wherever there is a distributed system

Communication Parameters: D, γ



Wired/wireless communication links

D = Event-response Delay, $D = min(D_i)$

$D \geq 1$ clock tick, i.e., bounded

γ = Communication Delay, $\gamma = max(\gamma_i)$

System Overview

- Synchronous message passing
- Fully connected graph with $K \geq 3F+1$ nodes
  (F = max number of simultaneous faults in the network)

Protocol Messages

- *Init* = {1, 0}
- *Echo* = Vector of locally time-stamped *Init* messages
- Messages arrive within time interval $[t+D, t+\gamma]$
- $D = min(D_i)$
- $\gamma = max(\gamma_i)$, for all i = 1..K

The Protocol

- Executes once every clock tick
- Based on initial coarse synchrony
- Triggered by another (primary) protocol
  - E.g., Symmetric-fault-tolerant protocol, *2015 IEEE Aerospace Conference*
- Integration of Primary and Secondary protocols is addressed in NASA/TM-2017-219638

What this protocol does

- Achieves <u>fine-grained synchrony</u> with optimum timing precision of 1 clock tick
  - Clock tick (no specific time units) ➔ Scalability
- Determines <u>network geometry</u> without initial knowledge of nodes' locations or distances between nodes
  - Accuracy is a function of clock precision

Applications

- Distributed networks
- GPS-Independent environment
    - Complementary/alternative to satellite systems
    - Last resort when GPS unavailable
- Wired / wireless network
- Dynamic network – shape and size
- Mobile network
- Local Positioning Systems (LPS)
- Localization – high accuracy, high-dynamic applications
- UAS in the NAS
- UAS Positioning / Navigation

    Ex. Crop dusting, search and rescue

The Protocol

if (LocalTimer = ψ)
    Broadcast Init
if (LocalTimer = ω + ψ)
    Broadcast Echo
if (LocalTimer = 2ω + ψ)
    Recover()
    Adjust()

- $\omega = \pi_{init} + \gamma$
- ψ = *ResetLocalTimerAt*

Recover()

- Recover Invalid *Init*
- Recover Invalid *Echo*

Adjust()

M = matrix of received messages at any N_x

row i = vector of locally time-stamped values received from N_i

column j = vector of reportedly received values from N_j

T = matrix of time-differences between nodes N_i and N_j

$$T(i,j) = (M(i,j) - M(j,i)) / 2 \qquad (1)$$

$$D_{ij} = C\,(M(i,j) + M(j,i)) / 2 \qquad (2)$$

D_{ij} will be actual distance between N_i and N_j upon synchrony

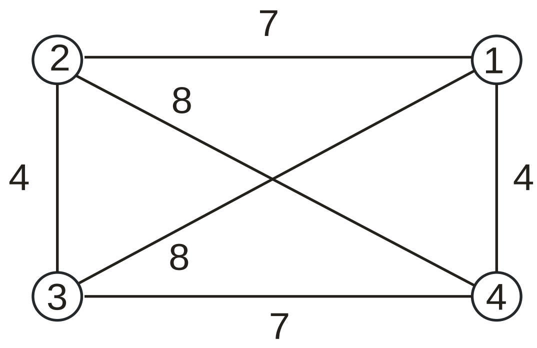


Table 1. Matrix *M*

16	21	32	18
9	16	22	16
0	2	16	5
6	16	25	16

Table 2. Matrix *T*

0	6	16	6
-6	0	10	0
-16	-10	0	-10
-6	0	10	0

$D_{12} = M(1,2) + M(2,1) / 2 = 15 * C$
$D_{13} = M(1,3) + M(3,1) / 2 = 16 * C$
$D_{14} = M(1,4) + M(4,1) / 2 = 12 * C$
$D_{23} = M(2,3) + M(3,2) / 2 = 12 * C$
$D_{24} = M(2,4) + M(4,2) / 2 = 16 * C$
$D_{34} = M(3,4) + M(4,3) / 2 = 15 * C$

Recover Invalid *Init*

- Link fault between N_i and N_j is recovered if there is valid data between N_i and N_j and N_x

- D_{if} is determined using trilateration and data in M

$$T(i,j) = T(i,x) - T(x,j) \tag{3}$$
$$M(i,j) = T(i,j) + D_{ij} \tag{4}$$

V = column f in M, i.e., $V = M(i,f) = valid$

Recover Invalid *Echo*

Repeat:

1. Determine D_{ij} using (2)
2. Realign: $V(i) = M(i, f) + T(j,i)$, for all i
3. Trilateration: Using V, determine when N_f had broadcast its message
   - Adjust V, $V(j) = V(j) - x$, for all j

Until (a or b)

a = Trilateration results in closest intersecting point

➔ Solution exists

b = Trilateration does not converge in π_{init}/x iterations

➔ Solution does not exist

If a solution exists, intersecting point is the time when N_f had broadcast its *Echo* and *xw* is amount of time took to reach the convergence point

Reconstruct $T(i,f)$

- $T(j,f) = xw$, where N_j is reference node used in Step 2
- $T(i,f) = T(j,f) - T(j,i)$, for all i and $i \neq j$
- $T(f,i) = -T(i,f)$, to preserve symmetry in T

Repair M using T and (1)

- $M(f,i) = M(i,f) - 2T(i,f)$, for all i

Find remaining distances D_{ij} between all nodes using (2)

Network geometry is now known

Adjust()

- Discard *F* values from both extremes and use midpoint
- $Adj = (RT + LT) / 2 = t_{MidPoint}$
- $LocalTimer = LocalTimer - Adj$

Proof of the Protocol

Lemma Correctness – *The protocol in slide 8 achieves optimum precision.*

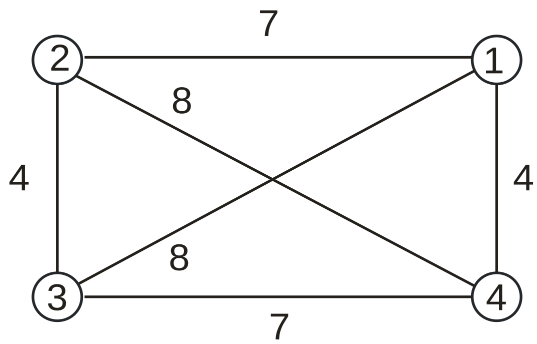


Table 1. Matrix M

16	21	32	18
9	16	22	16
0	2	16	5
6	16	25	16

$$D_{12} = M(1,2) + M(2,1) / 2 = 15 * C$$
$$D_{13} = M(1,3) + M(3,1) / 2 = 16 * C$$
$$D_{14} = M(1,4) + M(4,1) / 2 = 12 * C$$
$$D_{23} = M(2,3) + M(3,2) / 2 = 12 * C$$
$$D_{24} = M(2,4) + M(4,2) / 2 = 16 * C$$
$$D_{34} = M(3,4) + M(4,3) / 2 = 15 * C$$

Table 2. Matrix T

0	6	16	6
-6	0	10	0
-16	-10	0	-10
-6	0	10	0

Timeline of activities at N_1: 0 --- 6,6 -------- 16

Ignoring extremes, 0, 16, adjustment Amount = (6 + 6) / 2 = 6

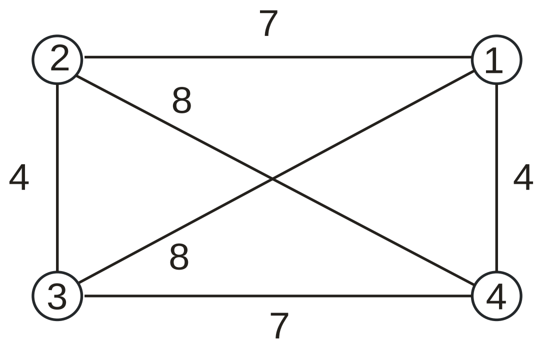


Table 3. Matrix M

8	7	8	4
7	8	4	8
8	4	8	7
4	8	7	8

Table 4. Matrix T

0	0	0	0
-0	0	0	0
-0	-0	0	-0
-0	-0	-0	0

$$D_{12} = M(1,2) + M(2,1) / 2 = 7 * C$$
$$D_{13} = M(1,3) + M(3,1) / 2 = 8 * C$$
$$D_{14} = M(1,4) + M(4,1) / 2 = 4 * C$$
$$D_{23} = M(2,3) + M(3,2) / 2 = 4 * C$$
$$D_{24} = M(2,4) + M(4,2) / 2 = 8 * C$$
$$D_{34} = M(3,4) + M(4,3) / 2 = 7 * C$$

Network geometry is known

Recover Invalid *Init*

Table 5. Matrix *M*

16	-	32	18
9	16	-	16
0	2	16	-
6	16	25	16

Table 6. Matrix *T*

0	-	16	6
-	0	-	0
-16	-	0	-
-6	0	-	0

$T(1,2) = T(1,4) - T(2,4) = 6 - 0 = 6$, $T(2,1) = -T(1,2) = -6$
$T(2,3) = T(1,3) - T(1,2) = 16 - 6 = 10$, $T(3,2) = -T(2,3) = -10$
$T(3,4) = T(1,4) - T(1,3) = 6 - 16 = -10$, $T(4,3) = -T(3,4) = 10$

M is restored using (1)
Network geometry is determined

For $K = 4$, $K-1 = 3$, simultaneous link faults are tolerated (recovered)

Recover Invalid *Echo*

Table 7. Matrix M

16	21	32	18
9	16	-	16
0	2	16	5
-	-	-	-

Table 8. Matrix T

0	6	16	-
-6	0	-	-
-16	-	0	-
-	-	-	-

$T(2,3) = T(1,3) - T(1,2) = 16 - 6 = 10$, $T(3,2) = -T(2,3) = -10$
From (1), $M(2,3) = 22$

Note N_4 did not broadcast *Echo* message to N_1
$V = M(1,4) = (18, 16, 5)$

Using V, D_{ij}, and trilateration, timing of N_4 in T is determined

M is subsequently restored using (1)
Network geometry is determined

Questions?