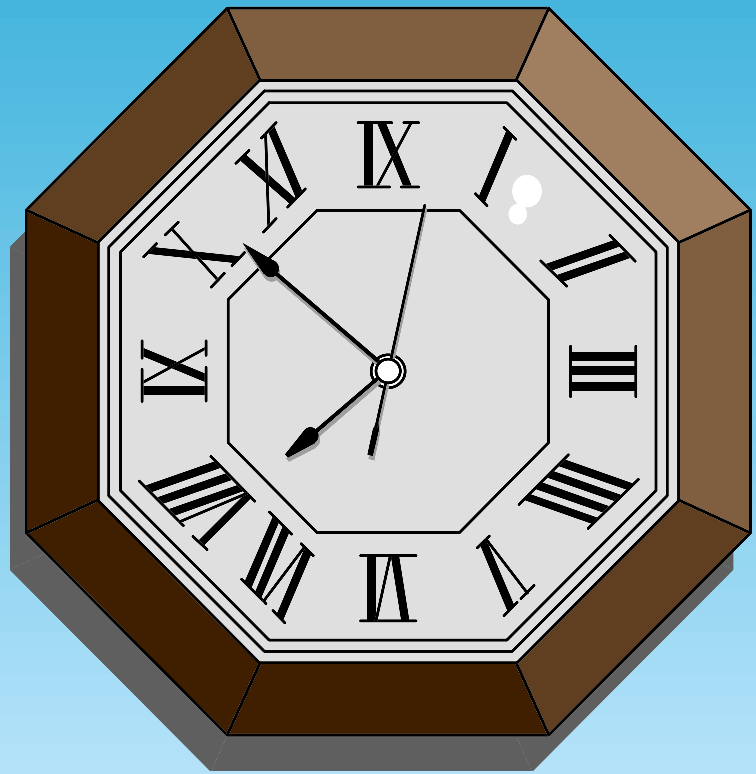
Marching to Many Different Drummers

Timothy Mann

Leslie Lamport



Dagstuhl Seminar on Time Services

March 1996

Background

- Joint work with Leslie Lamport in 1990
- Motivated by problems with DTS and NTP time services proposed for OSF DCE
- Too late to change DTS (or NTP) so much
- Work was never published

Nature of the work

- Algorithmic techniques and analysis: Raw material for building a time service
- Sketch of one possible time service
- No detailed design or implementation
- Theme: don’t discard useful information

Setting

- *Nodes* interconnected by a *network*
- Each node has a *local clock* with rate correct within $\pm\rho$
- *Time provider* nodes have an accurate time source
- A node may be *faulty*, including Byzantine faults
- Goal: The time service at each node provides
  - an interval containing *UT* (the correct time)
  - with fault tolerance

Basic concepts

- *Time datum:* A time interval I = [L, R] together with a failure predicate F
- *Failure predicate:* A boolean expression on node names; e.g., $A*B + C$
- A = 1 if node A has failed
- A time datum D = (I, F) asserts that $(UT \in I) \vee F$

Interpreting failure predicates

- *Degree* of a fp = min degree of its terms
  - $deg(A) = 1$
  - $deg(A + B*C) = 1$
  - $deg(A*B + B*C) = 2$
  - $deg(0) = \infty$ $\quad deg(1) = 0$
- = min node failures to make the fp true
- ≈ order of magnitude of probability that fp is true. Pun by substituting p into the fp.

Maintaining time data

- A time datum $D = ([L, R], F)$ gives information only about the *current* time
- If a node knows that $UT \in D$ now, after t seconds have passed on its local clock, it knows only that
  $UT \in ([L+t-\rho\, t, R+\mathrm{t}+\rho\, t], F)$
- So a node stores a datum as a triple (I, F, c) where c is a local clock reading

Transmitting time data

- If A sends B datum $D = [(L, R), F]$, then B knows that $UT \in$
  $D' = [(L+u, R+v), F + A + B]$
  - u, v = min, max transmission delay
  - A and B are added because if A or B is faulty, they may have corrupted the datum
- However, B will assume itself nonfaulty

Combining time data (1)

- Basic tautologies:

$$D_1 \wedge D_2 \Rightarrow (I_1 \cap I_2, F_1 + F_2)$$

$$D_1 \wedge D_2 \Rightarrow (I_1 \cup I_2, F_1 * F_2)$$

- Nonoverlapping intervals imply a failure; so we define *failure knowledge:*

$$FK(D_1) = 1$$

$$FK(D_1, D_2) = F_1 + F_2 \text{ if } I_1 \cap I_2 = \varnothing; \text{ 1 otherwise}$$

$$FK(D_1, \ldots, D_n) = \prod_{i,j} FK(D_i, D_j)$$

Relative degree

- Define the *degree of F relative to G:*
  $deg(F|G) = deg(F*G) - deg(G)$
- This is the minimum number of additional failures needed to make F true, given that G is true
- Interesting because we don't assume an absolute maximum number of faults in the system

Combining time data (2)

- How to combine many time data into one?
- Tradeoff between narrowness of resulting I and strength of resulting F
- We measure "strength" of F as $deg(F|FK)$ where the FK is that deduced from the input time data

Combining time data (3)

- We defined a class of combining functions *MLM[j,k]* that trade off width of *I* against strength of *F*
  - Choose the *jth*-best left endpoint and *kth*-best right endpoint for *I*
  - Compute the corresponding *F* and *FK*
- Details in the paper

Using the techniques

- When asked the time:
  - Combine available data to get the answer
  - Don't necessarily discard the individual data

When do nodes exchange data?

- We leave this open; different time service designs result from different choices.
- One sketch of an idea:
  - Nodes are assigned to a hierarchy
  - Time providers at the top, others below
  - Data flows from higher to lower nodes
  - Periodic broadcast or request/response

When do nodes discard data?

- Again open, but here are some hints:
  - Stale: wider *I* with same *F*
  - Gone too far: similar *I*, but *F* a superset
  - Little information: *I* very wide
  - Probably wrong: *FK* says *F* is true (or likely)
  - Use *MLM* and keep only the combined data

Other topics in the paper

- What to do when a failed node recovers
- Effect of Byzantine failures
- Fuller implementation sketch
- Space/time cost estimate for the above

What next?

- We should publish this!
  - The concepts may be useful to others in future work and perhaps in analyzing existing work
  - We expect to do so, at least as a tech report
  - We might file for a US patent
- An implementation?
  - Could be based on the sketch in the paper
  - Might be a good summer intern project