Verification of Fault-Tolerant Clock Synchronization Algorithms (Benchmark Proposal)

Sergiy Bogomolov[1], Christian Herrera[2] and Wilfried Steiner[3]

IST Austria[1],
University of Freiburg[2],
TTTech Computertechnik AG[3]

April 11th, 2016

Motivation – TTEthernet

TTEthernet

- implementation of the *Ethernet* standard which complies with time-critical, deterministic and safety-critical real-time requirements.

Motivation – TTEthernet

TTEthernet

- implementation of the *Ethernet* standard which complies with time-critical, deterministic and safety-critical real-time requirements.
- used in commercial hardware and software products, e.g. the avionics of the *Orion Space Program*.

Motivation – TTEthernet

TTEthernet

- implementation of the *Ethernet* standard which complies with time-critical, deterministic and safety-critical real-time requirements.
- used in commercial hardware and software products, e.g. the avionics of the *Orion Space Program*.
- assumes a global time base for tolerating faulty behavior of safety critical systems, i.e.
  - any two logical clocks of two components must read the same values at any time (*precision*). The precision is ensured by a clock synchronization algorithm.

Our Goal

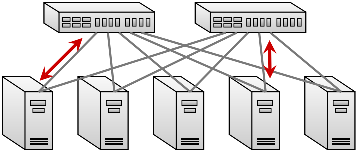
We propose a *benchmark* which can be used as a basis for:

- verifying the *precision* of the clock synchronization algorithm.
- implementing optimization techniques for verification purposes, e.g. detection and reduction of *quasi-dependent variables*.
- verifying other clock synchronization algorithms, e.g. *interactive convergence algorithm* and *byzantine clock synchronization*.

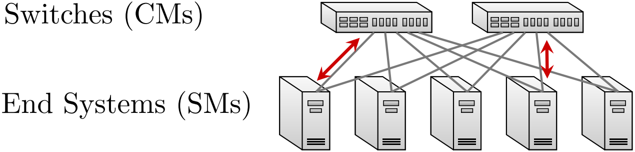
Fault-Tolerant Configuration

Switches (CMs)

End Systems (SMs)

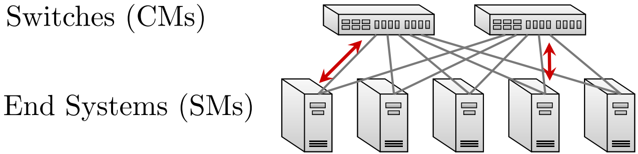


TTEthernet's Clock Synchronization Algorithm



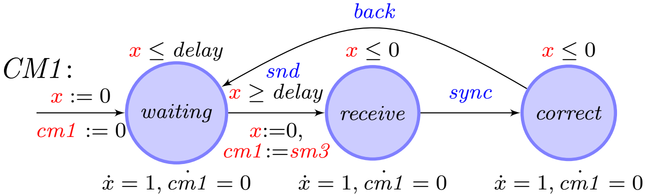
1. SMs send the current value of their clocks to each CM. Then each CM obtains the median of the values sent by all SMs.

TTEthernet's Clock Synchronization Algorithm

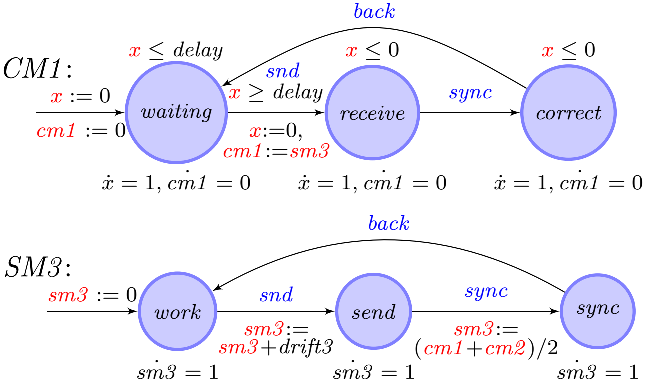


1. SMs send the current value of their clocks to each CM. Then each CM obtains the median of the values sent by all SMs.
2. CMs send the mentioned result to each SM. Then each SM updates its clock by using the median of the values sent by all CMs.

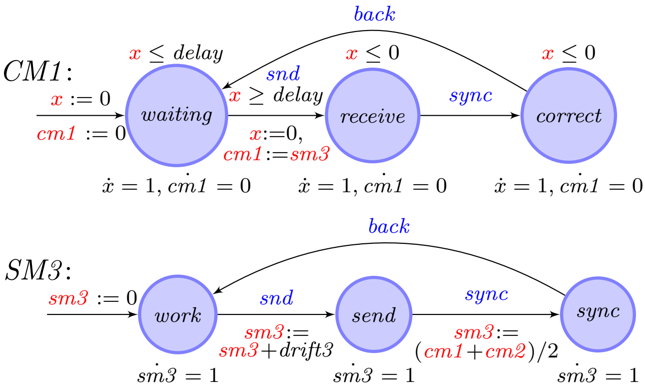
Benchmark – Network of Hybrid Automata $\mathcal{N}$



Benchmark – Network of Hybrid Automata $\mathcal{N}$



Benchmark – Network of Hybrid Automata $\mathcal{N}$



Precision:

$$\forall i \neq j \in \mathbb{N} \bullet \; sm_i > sm_j \implies sm_i - sm_j \leq 2 * maxdrift$$

Optimization for Verification Purposes

- We detect and reduce *quasi-dependent variables* in $\mathcal{N}$.
- Given two variables x and y, x quasi-depends on y via function f, if and only if $x = f(y)$ in all runs of $\mathcal{N}$ and at all points in time, except when x and y are updated by discrete actions.

Optimization for Verification Purposes

- We detect and reduce *quasi-dependent variables* in $\mathcal{N}$.
- Given two variables x and y, x quasi-depends on y via function f, if and only if $x = f(y)$ in all runs of $\mathcal{N}$ and at all points in time, except when x and y are updated by discrete actions.
- We obtain equivalence classes of quasi-dependent variables.
- We use only the representative clock of each class in a transformed network which satisfies the same properties as the original network.

Experiments

Components	Clocks		Run Time (sec.)	
	$\mathcal{N}$	$\mathcal{N}'$	$\mathcal{N}$	$\mathcal{N}'$
$5+2$	7	2	31.32	11.96
$20+2$	22	2	124.08	12.11
$30+2$	32	2	201.21	12.57

Note the following:

- $\mathcal{N}'$ is the network output by our detection and reduction of quasi-dependent variables approach.
- $\mathcal{N}'$ uses only one representative clock for all CMs, and one representative clock for all SMs.

Open Problems of the Benchmark

Note the following:

1. We assume that the rate of each clock is 1. In practice each clock of a SM may tick slower or faster after n time units.
2. A *rate correction algorithm* in TTEthernet can correct the rates of those clocks.
3. Remains unclear how to detect and reduce quasi-dependent variables in benchmarks where clocks tick slower or faster than 1 after n time units.

Thanks for your attention.