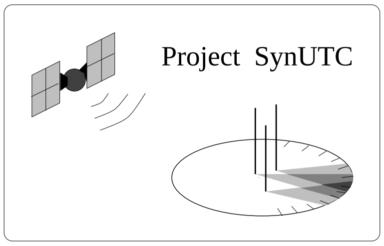
Interval-based Clock Synchronization

Ulrich Schmid

July 4, 1997



This research (partly joint work with Klaus Schossmaier) was performed in the framework of our project SynUTC, which is supported by the Austrian Science Foundation (FWF) under grant P10244-ÖMA, the Austrian START-Programme Y41-MAT, the BMWV under contract Zl.601.577/2-IV/B/9/96, and the OeNB under grant no. 6454.

The Interval-based Paradigm

(Marzullo 1984; Lamport 1987; OSF 1992; Schmid 1995; Schmid and Schossmaier 1997; Schmid 1997a; Schmid 1997b)

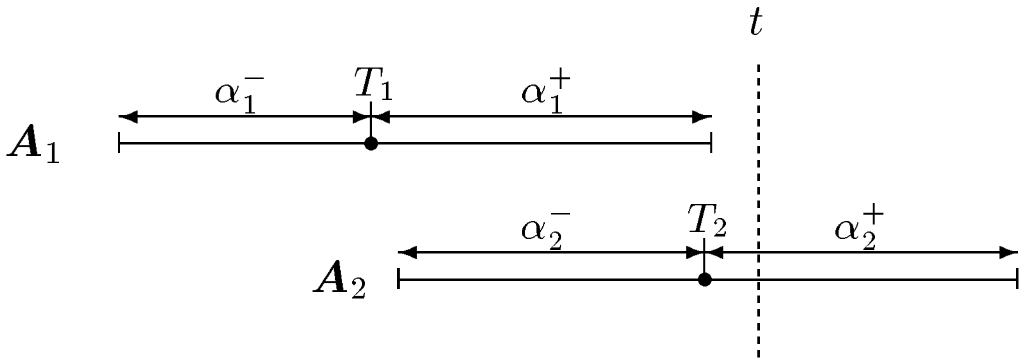
Replace ordinary clocks $T = C(t)$ by *interval clocks*

$$\boldsymbol{C}(t) = [C(t) - \alpha^-(t), C(t) + \alpha^+(t)]$$

incorporating

- an ordinary local (quartz) clock $C(t)$
- an interval of accuracies $\boldsymbol{\alpha}(t) = [-\alpha^-(t), \alpha^+(t)]$

Logical time information representing real-time t is hence an (asymmetric) *accuracy interval* $\boldsymbol{A} = [T \pm \boldsymbol{\alpha}]$:



Accuracy intervals

- are called *accurate* iff $t \in \boldsymbol{A}$
- bound Accuracy w.r.t. real-time via α^-, α^+
- are called *compatible* iff they represent the same t

Basic Algorithm

Synchronous system assumptions

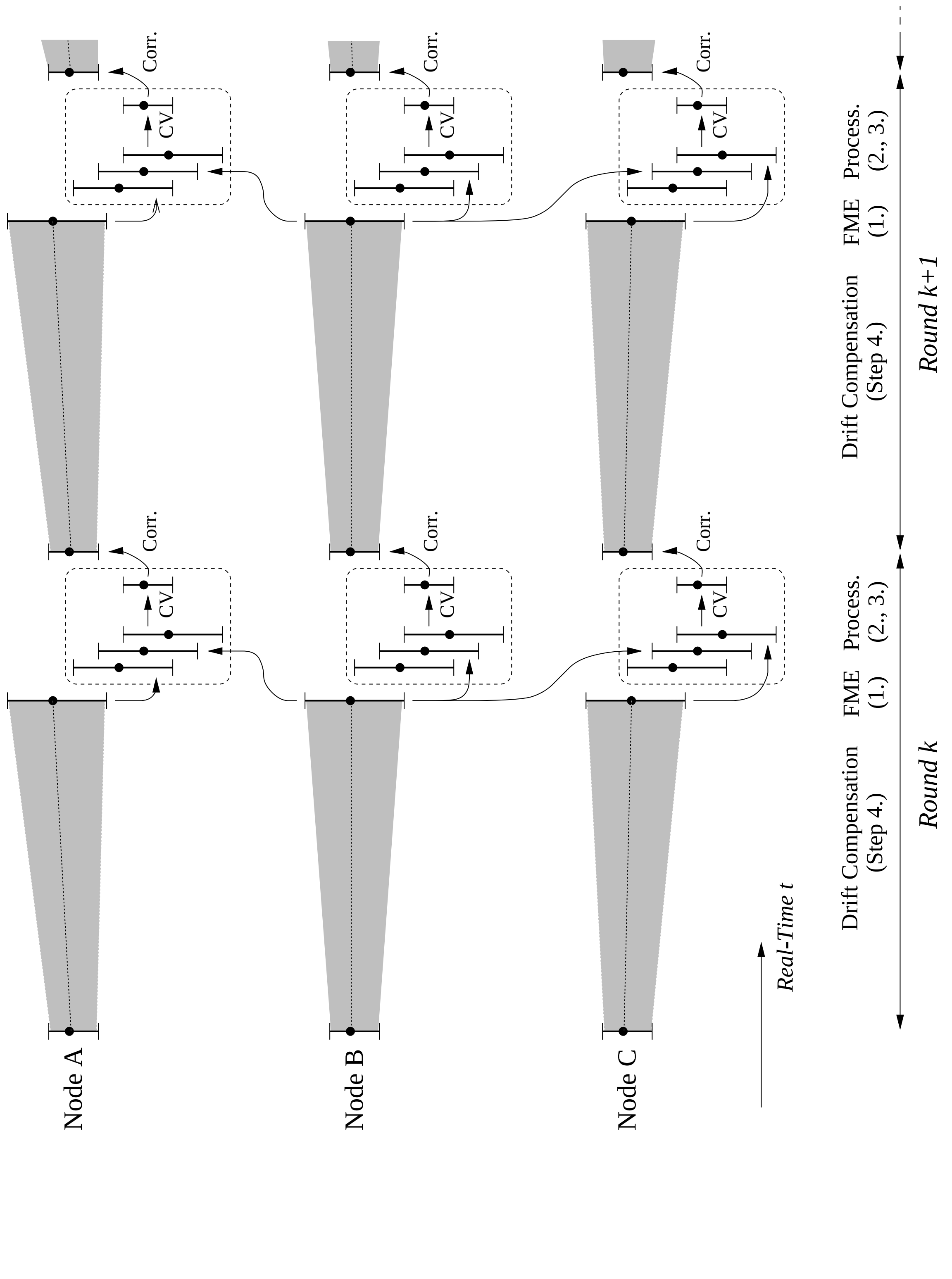
- Every non-faulty node p executes the clock synchronization algorithm within γ_p seconds
- Every node p is equipped with a local interval clock $\boldsymbol{C}_p(t)$ with (inverse) rate deviation bound $\boldsymbol{\rho}_p$, granularity G, and rate adjustment uncertainity $\boldsymbol{u}_p$
- Communication between non-faulty nodes p and q experiences transmission delay $\delta'_{pq} \in [\delta_{pq} \pm \boldsymbol{\varepsilon}_{pq}]$, broadcast operation delay $\omega_{\max}$, broadcast latency $\lambda_{\max}$, and accuracy resolution loss G_A
- Hybrid fault model, covering crash, benign, and arbitrary failures (depending on convergence function)

Round-based algorithm: When local clock $C_p(t)$ reads kP, $k \geq 1$, every node p performs

1. Broadcast $\boldsymbol{C}_p$ to all nodes in the system (FME)
2. Processing of received accuracy intervals to make them all compatible ($\Rightarrow$ *drift* and *delay compensation*)
3. Computing and subsequently applying a correction value for $\boldsymbol{C}_p(t)$ ($\Rightarrow$ *convergence function*)
4. Local maintenance of $\boldsymbol{C}_p(t)$ up to the next broadcast ($\Rightarrow$ *drift compensation*).

Overall Execution Example

Since local clocks are synchronized, the algorithm yields the following global execution of a three-node system:



Building Blocks: Drift Compensation

Problem:

- Given an accurate interval $\boldsymbol{I} = \boldsymbol{I}(t) = [T \pm \boldsymbol{\alpha}]$,
- "drag" $\boldsymbol{I}$ by means of local clock to time t'.
- What about the result $\boldsymbol{I}' = \boldsymbol{I}'(t')$?

Local clock $C(t)$ not perfect but can drift from real-time at most by inverse rate deviation bound $\boldsymbol{\rho}$.

Deterioration required to maintain accurateness

$\Rightarrow$ "blow up" $\boldsymbol{I}$'s reference point to an interval accounting for maximum error due to drift

$\Rightarrow$ enlarge α^- and α^+ of $\boldsymbol{I}$ accordingly

Additional issues

- Clock granularity G
- Rate adjustment uncertainity $\boldsymbol{u}$

Building Blocks: Delay Compensation

Problem:

- Given accurate $\boldsymbol{I} = \boldsymbol{I}(t) = [T \pm \boldsymbol{\alpha}]$ at node p,
- sent from node p to q, experiencing some transmission delay $\delta' \in [\delta \pm \boldsymbol{\varepsilon}]$.
- What about the result $\boldsymbol{I}'' = \boldsymbol{I}''(t')$ at node q?

Uncertainity in transmission delay impairs accurateness of $\boldsymbol{I}''$.

Delay compensation required for maintaining accurateness despite of variable transmission delays

- shifting the sent $\boldsymbol{I}$ by δ
- "blowing up" the shifted interval's reference point to $\boldsymbol{\varepsilon}$

Additional issue to be considered

- Interval of accuracies usually not transmitted with full resolution $\Longrightarrow$ accuracy resolution loss G_A

Building Blocks: Convergence Function

Problem:

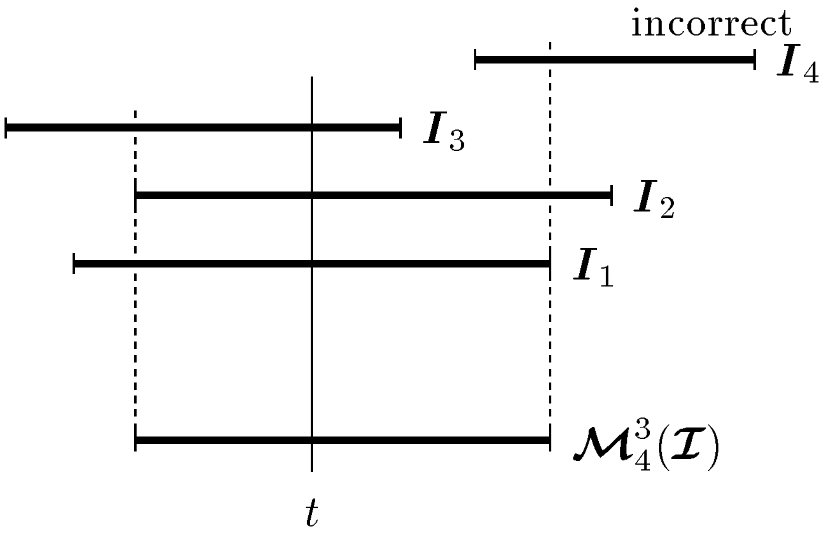
- Given a set $\boldsymbol{\mathcal{I}}_p$ of (received and preprocessed) accuracy intervals $\boldsymbol{I}_p^s$ all representing t,
- some of those intervals being faulty.
- How to compute an improved accuracy interval for the local interval clock?

Since all non-faulty $\boldsymbol{I}_p^s$ contain t, it is quite natural to consider some kind of fault-tolerant intersection here.

The optimal choice w.r.t. the length of the resulting accuracy interval is

Definition 1 (Marzullo's Function $\boldsymbol{\mathcal{M}}$) *Given set* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ *of compatible intervals representing* t, $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}})$ *is the largest interval whose endpoints intersect at least* $n-f$ *of the* $\boldsymbol{I}_j$*'s.*

Example of Marzullo's Function



Observations

- The (unknown) t cannot lie between the end of I_3 and the beginning of I_4 covered by $\mathcal{M}_4^3$
- However, it is impossible to decide whether t lies left or right of this region
- $\mathcal{M}_4^3(\mathcal{I}) \subseteq I_1 \cap I_2$, hence cannot be larger than those input intervals

Note that

- Accuracy intervals could grow in each round
- Maintaining accuracy does not need reference points

Generic Convergence Function $\mathcal{CV}$

Our framework admits any interval-based convergence function that conforms to our generic $\mathcal{CV}$.

Its application $\boldsymbol{C}_p = \mathcal{CV}(\boldsymbol{\mathcal{I}}_p)$ to a set of intervals $\boldsymbol{\mathcal{I}}_p$ conforming to the particular fault model must

- provide an accurate interval: $t \in \boldsymbol{C}_p$
- enhance/preserve accuracy: $|\boldsymbol{C}_p| \leq |\boldsymbol{I}_p^s|$
- enhance precision:

$$|\mathrm{ref}(\boldsymbol{C}_p) - \mathrm{ref}(\boldsymbol{C}_q)| < |\mathrm{ref}(\boldsymbol{I}_p^s) - \mathrm{ref}(\boldsymbol{I}_q^s)|$$

Note that

- "Naive" reference point setting (centerpoint) cannot guarantee precision enhancement if $\mathcal{M}$ is used
- Some explicit means must be provided for achieving precision enhancement here

Existing convergence functions

- Orthogonal Accuracy Convergence Function $\mathcal{OA}$
- Optimal Precision Convergence Function $\mathcal{OP}$

Interval-based Analysis

The analysis of our basic algorithm is based upon keeping track how accuracy intervals evolve over rounds. The same principle is utilized for precision analysis, by replacing

- Accuracy intervals by precision intervals
- Real-time by a suitable internal global time

Results: Worst case bounds for

- Accuracies α^-, α^+ after k rounds
- Precision $\pi_{\max}$
- Difference of resynchronization real-times at any two nodes
- Maximum clock correction value
- Global rate

that

- are expressed in terms of the characteristic parameters of the convergence function,
- incorporate all the parameters provided for in our system model,
- are provided both for instantaneous correction and continuous amortization

Specific results

Orthogonal Accuracy Algorithm OA

- OA performs like fault-tolerant midpoint w.r.t. precision, hence is suboptimal:
  - precision $\approx 5\varepsilon + 4P\rho$
  - global rate $\approx \rho + \varepsilon/P$
  - maximum clock correction $\approx 5\varepsilon + 4P\rho$
- Clock granularity G and rate adjustment uncertainity $\boldsymbol{u}$ seriously affect worst case precision by $4G+10u$
- Unlike $\mathcal{M}$, OA is not optimal w.r.t. length of accuracy intervals due to excessive reference point setting

Optimal Precision Algorithm OP

- OP provides optimal precision like the algorithm of (Fetzer and Cristian 1995):
  - precision $\approx 4\varepsilon + 4P\rho$
  - global rate $\approx \rho$
  - maximum clock correction $\approx 2P\rho$
- Slightly less effects of clock granularity and rate adjustment uncertainity $3G + 9u$
- Less suboptimal w.r.t. length of accuracy intervals