10 Synchronizing by Approximate Agreement

Chapter Contents

10.1 Overview 149
10.2 Approximate Agreement 149
10.3 A Variant of the Lynch-Welch Algorithm 153

In a previous chapter (Chapter 9), we've seen how to achieve a skew of $O(d)$ in a system of n fully connected nodes with $f < n/3$ Byzantine faults. We've also seen that we can't do any better in terms of the number of faults that can be tolerated. So let's ask our usual question: Is this skew bound (asymptotically) optimal or can we do better? Already in a fault-free system, we know that we can't beat $\Omega(u + (\vartheta - 1)d)$. But can this bound be attained in the presence of faults?

In this chapter we introduce a target function that iteratively reduces the range of values the participating nodes hold. We make use of this technique to improve that skew.

10.1 Overview

10.2 Approximate Agreement

The answer is provided by leveraging techniques for the task of *approximate agreement.* For this problem, we assume the (convenient) abstraction of a synchronously operating system. For simplicity, we make no use of the detailed state machine description given in Definition 6.1. Hiding such details, for the purposes of this chapter synchronous operation can be described as follows:

A synchronous execution proceeds in *synchronous rounds.* At the start of the execution, each node receives an input (whose type depends on the task at hand). In each round,

1. nodes perform local computations,

2. send messages to their neighbors in the network graph,
3. receive the messages of their neighbors, and
4. (optionally) may compute an output value and terminate (i.e., stop executing the other steps in future rounds).

Note that a synchronous execution of a deterministic algorithm is fully determined by the input values and the (arbitrary) messages sent by faulty nodes.

This model provides a very clean abstraction for describing the tool we would like to use.

Definition 10.1 (Approximate Agreement)**.** *Each node $v \in V$ is given an input value $r_v \in \mathbb{R}$. Given a constant $\varepsilon > 0$, the task is to generate output values $o_v \in \mathbb{R}$ so that*

agreement: $\max_{v,w \in V_g}\{o_v - o_w\} \leq \varepsilon$,

validity: $\forall v \in V_g : \min_{w \in V_g}\{r_w\} \leq o_v \leq \max_{w \in V_g}\{r_w\}$, *and*

termination: *each $v \in V_g$ determines its output o_v and terminates within a finite number of rounds.*

Once we solved approximate agreement in this abstract model, we will employ it to agree on when the nodes should generate clock pulses, i.e., solve the pulse synchronization problem with it. The simulation of the synchronous algorithm (cf. Sections 9.3.2 and 9.6.6) and maintaining a small skew will go hand in hand!

Solving Approximate Agreement

Algorithm 12 An iterative step of the Approximate Agreement at node $v \in V_g$ (with synchronous message exchange).

```
1: broadcast $r_v$ to all nodes (including self)  // node $v$ is given input value $r_v$
2: receive $\hat{r}_{wv}$ from each node $w$
     ($\hat{r}_{wv} := r_v$ if no message with correct type of content from $w$ received)
3: $S_v \leftarrow \{\hat{r}_{wv} \mid w \in V\}$
4: $o_v \leftarrow \frac{S_v^{(f+1)} + S_v^{(n-f)}}{2}$
5: return $o_v$
```

Since minimizing the maximum difference between correct nodes' values is our goal, the following definition provides a useful shorthand.

Definition 10.2 (Diameters of Vectors)**.** *Denote by $\vec{r}$ the $|V_g|$-dimensional vector of correct nodes' inputs, i.e., $(\vec{r})_v = r_v$ for $v \in V_g$. Denote by $r^{(k)}$, $k \in \{1, \ldots, |V_g|\}$, the k-th entry when ordering the entries of $\vec{r}$ ascendingly.*

The diameter $\|\vec{r}\|$ *of* $\vec{r}$ *is the difference between the maximum and minimum components of* $\vec{r}$*. Formally,*

$$\|\vec{r}\| := r^{(|V_g|)} - r^{(1)} = \max_{v \in V_g}\{r_v\} - \min_{v \in V_g}\{r_v\}.$$

We will use the same notation for other values, e.g. $\vec{o}$*,* $o^{(k)}$*,* $\|\vec{o}\|$*, etc.*

For simplicity, we assume that $|V_g| = n - f$ in the following; all statements can be adapted by replacing $n - f$ with $|V_g|$ where appropriate. As usual, we require that $3f < n$.

Intuitively, Algorithm 12 discards the smallest and largest f values each to ensure that values from faulty nodes cannot cause outputs to lie outside the range spanned by the correct nodes' values. Afterwards, o_v is determined as the midpoint of the interval spanned by the remaining values. Since $f < n/3$, i.e., $n - f \geq 2f + 1$, the median of correct nodes' values is part of all intervals computed by correct nodes. From this, it is easy to see that $\|\vec{o}\| \leq \|\vec{r}\|/2$. We now prove these properties.

Lemma 10.3.

$$\forall v \in V_g : r^{(1)} \leq o_v \leq r^{(n-f)}.$$

Proof. As there are at most f faulty nodes, for $v \in V_g$ we have that

$$S_v^{(f+1)} \geq \min_{w \in V_g}\{\hat{r}_{wv}\} = r^{(1)}.$$

Analogously, $S_v^{(n-f)} \leq r^{(n-f)}$. We conclude that

$$r^{(1)} \leq S_v^{(f+1)} \leq \frac{S_v^{(f+1)} + S_v^{(n-f)}}{2} = o_v \leq S_v^{(n-f)} \leq r^{(n-f)}.$$ □

Lemma 10.4. $\|\vec{o}\| \leq \|\vec{r}\|/2$.

Proof. Since $f < n/3$, we have that $n - f \geq 2f + 1$. Hence, for all $v \in V_g$,

$$r^{(1)} \leq S_v^{(f+1)} \leq r^{(f+1)} \leq S_v^{(2f+1)} \leq S_v^{(n-f)} \leq r^{(n-f)}.$$

For any $v, w \in V_g$, it follows that

$$\begin{aligned} o_v - o_w &= \frac{S_v^{(f+1)} - S_w^{(f+1)} + S_v^{(n-f)} - S_w^{(n-f)}}{2} \\ &\leq \frac{r^{(f+1)} - r^{(1)} + r^{(n-f)} - r^{(f+1)}}{2} = \frac{r^{(n-f)} - r^{(1)}}{2} \\ &= \frac{\|\vec{r}\|}{2}. \end{aligned}$$

As $v, w \in V_g$ were arbitrary, this yields $\|\vec{o}\| \leq \|\vec{r}\|/2$. □

Applying this approach inductively yields a straightforward algorithm provided an upper bound $R \geq r^{(|V_g|)} - r^{(1)}$ is known.

Theorem 10.5 (Approximate Agreement)**.** *Applying Algorithm 12 iteratively (using the output of one step as input to the next) for $\lceil \log(R/\varepsilon) \rceil$ steps solves approximate agreement.*

Proof. Agreement readily follows from inductive application of Lemma 10.4. Applying Lemma 10.3 inductively shows validity. By construction, all nodes terminate after $\lceil \log(R/\varepsilon) \rceil$ synchronous rounds. □

Modifications for the Pulse Synchronization Problem

In our setting, we will not be able to guarantee exact communication of clock values. Accordingly, we slightly modify the communication model. More specifically, at certain times, nodes will need estimates of each other's logical clock values. Node v will use its estimate of w's clock value as approximation of the "input" r_w of $w \in V$. Thus, instead of receiving $\hat{r}_{wv} = r_w$ from $w \in V$, v will receive $\hat{r}_{wv}$ satisfying

$$r_w \leq \hat{r}_{wv} \leq r_w + \delta$$

for some $\delta > 0$ that we will determine later. As shifting the values $\hat{r}_{wv}$ in Algorithm 12 by less than δ will affect the outputs by less than δ, we obtain the following corollary to Lemmas 10.3 and 10.4. See Figure 10.1 for a visualization.

Corollary 10.6. *With the above modification to the communication model, Algorithm 12 guarantees*

(i) $\forall v \in V_g : r^{(1)} \leq o_v \leq r^{(n-f)} + \delta$ *and*
(ii) $\|\vec{o}\| \leq \|\vec{r}\|/2 + \delta$.

Now all we need to do is to gather estimates, use Algorithm 12 to determine adjustments to the logical clocks, and iterate.

The algorithm is now constructed as follows. Assuming some bound $H \geq \max_{v \in V_g}\{H_v(0)\}$ on the skew at initialization, nodes generate their first pulse at local time H. This marks the (local) start of the first round. Then they wait until they can be sure that all nodes have generated their pulse. At the respective hardware time, they transmit an empty message — no content is needed, as the local time when the message is sent is hardwired into the algorithm. Then nodes wait until the local time when all such messages from correct nodes are certainly received and compute their estimates of the relative clock differences to other nodes. Finally, they apply Algorithm 12 to compute an adjustment to the (local) starting time of the next round. This ensures bounded skew for the

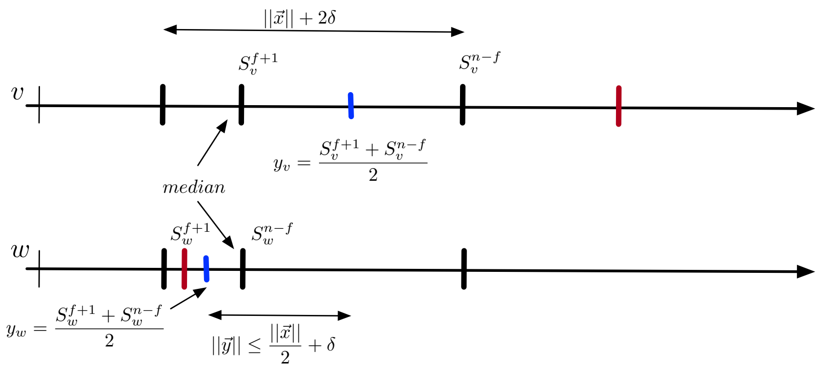
$||\vec{x}|| + 2\delta$
S_v^{f+1}
S_v^{n-f}
v
$y_v = \frac{S_v^{f+1} + S_v^{n-f}}{2}$
$median$
S_w^{f+1}
S_w^{n-f}
w
$y_w = \frac{S_w^{f+1} + S_w^{n-f}}{2}$
$||\vec{y}|| \le \frac{||\vec{x}||}{2} + \delta$

Figure 10.1
An execution of Algorithm 12 at nodes v and w of a system consisting of $n = 4$ nodes. There is a single faulty node and its values are indicated in red. Note that the ranges spanned by the values received from correct nodes are *almost* identical; the difference originates in the perturbations of up to δ.

next pulse and thus also the starting times of the next round. From there, the process is iterated.

10.3 A Variant of the Lynch-Welch Algorithm

Algorithm 13 is phrased in a parametrized fashion suitable for the analysis. This means that we assume a skew bound of $\mathcal{S}$ to hold on initialization, an error bound δ on the logical clock estimates nodes compute of each other, and a nominal round duration of T. We then determine valid choices for these parameters from the analysis, where we need to determine δ depending on how the estimates are computed.

"Rounds" of the algorithm simulate the synchronous operation assumed in the approximate agreement problem, where each iteration of the loop simulates one synchronous round. For this to work as intended, two requirements need to be met in each round:

(i) Messages sent by correct nodes are received at all correct nodes after starting the round and before they compute their clock adjustment, i.e., during $[p_{v,r}, \tau_{v,r}]$.
(ii) T is large enough to accomodate the adjustments of the next pulse times. This means to ensure that $H_v(\tau_{v,r}) \le h_v(\tau_{v,r}) + \Delta_v(\tau_{v,r}) + T$, i.e., that $H_v(\tau_{v,r}) \le H_v(p_{v,r}) + \Delta + T$, where Δ is the value determined in Line 12 of the r-th iteration of the loop.

Algorithm 13 Lynch-Welch pulse synchronization algorithm, the code for a node $v \in V_g$. $\mathcal{S}$ denotes a (to-be-determined) upper bound on $\|\vec{p}_r\|$ for each $r \in \mathbb{N}_{>0}$ and T is the nominal round duration and it needs to be specified how Line 9 is implemented.

```
1:  wait until getH() = $\mathcal{S}$                         // $H_w(0) \in [0, \mathcal{S})$ for all $w \in V$
2:  for all round $r \in \mathbb{N}$ do
3:      generate $r$-th pulse
4:      $h \leftarrow$ getH()
5:      wait until getH() = $h + \vartheta\mathcal{S}$                  // all nodes are in round $r$
6:      broadcast empty message to all nodes (including self)
7:      wait until getH() = $h + (\vartheta^2 + \vartheta)\mathcal{S} + \vartheta d$     // denote this time by $\tau_{v,r}$
                                           // correct nodes' messages should have arrived
8:      for each node $w \in V$ do
9:          compute $\Delta(w) \in [p_{w,r} - p_{v,r}, p_{w,r} - p_{v,r} + \delta]$
                                           // denote $p_r := \max_{w \in V_g}\{p_{w,r}\}$
10:     end for
11:     $S \leftarrow \{\Delta(w) \mid w \in V\}$ (as multiset, i.e., values may repeat)
12:     $\Delta \leftarrow \left(S_v^{(f+1)} + S_v^{(n-f)}\right)/2$
13:     wait until getH() = $h + \Delta + T$
14: end for
```

If these properties are satisfied in round r, we will say that *round r is executed correctly.* We will show that this holds for all $r \in \mathbb{N}$ inductively, where the induction hypothesis is that $\|\vec{p}_r\| \leq \mathcal{S}$; this simulatenously shows that the algorithm has a small skew! For $r = 1$, this is immediate from our assumption on the initial hardware clock values.

Lemma 10.7. *Suppose that $T \geq (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d$ and*

$$\mathcal{S} \geq \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta}.$$

Moreover, assume that for $r \in \mathbb{N}$ it holds that all prior rounds have been executed correctly, and that $\|\vec{p}_r\| \leq \mathcal{S}$. Then

(i) round r is executed correctly,

(ii) $(T - \mathcal{S})/\vartheta \leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} \leq T + \mathcal{S} + \delta$, and

(iii) $\|\vec{p}_{r+1}\| \leq \mathcal{S}$.

Proof. By assumption, no messages sent by correct nodes in rounds $r' < r$ are received in round r. Consider the message $v \in V$ sends after entering round r. It is sent no earlier than local time $h_v(p_{v,r}) + \vartheta\mathcal{S} = H_v(p_{v,r}) + \vartheta\mathcal{S}$, and hence no earlier than time

$\frac{dH_v}{dt} \leq \vartheta$

$\|\vec{p}_r\| \leq \mathcal{S}$

$$\begin{aligned} H_v^{-1}(H_v(p_{v,r}) + \vartheta\mathcal{S}) &\geq p_{v,r} + \mathcal{S} \\ &\geq \max_{w \in V_g}\{p_{w,r}\}. \end{aligned}$$

It is received by time

$\frac{dH_v}{dt} \geq 1$

$\|\vec{p}_r\| \leq \mathcal{S}$

$$\begin{aligned} H_v^{-1}(H_v(p_{v,r}) + \vartheta\mathcal{S}) + d &\leq p_{v,r} + \vartheta\mathcal{S} + d \\ &\leq \min_{w \in V_g}\{p_{w,r}\} + (\vartheta + 1)\mathcal{S} + d. \end{aligned}$$

As $\tau_{w,r} \geq p_{w,r} + (\vartheta + 1)\mathcal{S} + d$ for all $w \in V_g$ (again because $\frac{dH_v}{dt} \leq \vartheta$), this shows part (i) of the requirements for correct execution of round r.

Concerning part (ii), assume that Algorithm 12 would be executed with inputs $p_{v,r}$ for each $v \in V_g$. Equivalently to executing the algorithm, each node could shift all the measured values by subtracting $p_{v,r}$ and shift the return value back by adding $p_{v,r}$. Thus, Algorithm 13 carries out this procedure, except that the inputs suffer from an additional error of (up to) δ. This is accounted for by Corollary 10.6, whose first statement then shows that

$\|\vec{p}_r\| \leq \mathcal{S}$

$$\Delta_v(\tau_{v,r}) + p_{v,r} \geq \min_{w \in V_g}\{p_{w,r}\} \geq p_{v,r} - \mathcal{S}, \tag{10.1}$$

i.e., $\Delta_v(\tau_{v,r}) \geq -\mathcal{S}$. Therefore,

(10.1)

def. of $\tau_{v,r}$

$T \geq \vartheta d + (\vartheta^2 + \vartheta + 1)\mathcal{S}$

$$\begin{aligned} H_v(p_{v,r}) + \Delta_v(\tau_{v,r}) + T &\geq H_v(p_{v,r}) - \mathcal{S} + T \\ &= H_v(\tau_{v,r}) - (\vartheta^2 + \vartheta + 1)\mathcal{S} - \vartheta d + T \\ &\geq H_v(\tau_{v,r}). \end{aligned}$$

This shows part (ii) of the requirements for correct execution of round r, establishing the part (i) of the claim of the lemma for round r. In particular, the times $p_{v,r+1}$, $v \in V_g$, are well-defined, as all correct nodes generate pulse $r + 1$ upon starting the next iteration of the loop.

Moreover, for each $v \in V_g$ it follows that

Line 13

(10.1)

$$\begin{aligned} H_v(p_{v,r+1}) - H_v(p_{v,r}) &= T + \Delta_v(\tau_{v,r}) \\ &\geq T - \mathcal{S}, \end{aligned}$$

implying that $p_{v,r+1} - p_{v,r} \geq (T - \mathcal{S})/\vartheta$ and thereby showing the lower bound for part (ii) of the claim of the lemma for round r. To show the upper bound, we apply statement (i) of Corollary 10.6 as above, but use the upper bound on the output values. We get that

$$\Delta_v(\tau_{v,r}) + p_{v,r} \leq \max_{w \in V_g}\{p_{w,r}\} + \delta \leq p_{v,r} + \mathcal{S} + \delta, \tag{10.2}$$

i.e., $\Delta_v(\tau_{v,r}) \leq \mathcal{S} + \delta$.

$$\begin{aligned} H_v(p_{v,r+1}) - H_v(p_{v,r}) &= T + \Delta_v(\tau_{v,r}) && \text{Line 13} \\ &\leq T + \mathcal{S} + \delta. && (10.2) \end{aligned}$$

For the choice $v = \operatorname{argmin}_{w \in V_g}\{p_{w,r}\}$, this yields

$$\begin{aligned} \min_{w \in V_g}\{p_{w,r+1}\} - \min_{w \in V_g}\{p_{w,r}\} &\leq p_{v,r+1} - p_{v,r} && \text{choice of } v \\ &\leq H_v(p_{v,r+1}) - H_v(p_{v,r}) && \tfrac{dH_v}{dt} \geq 1 \\ &\leq T + \mathcal{S} + \delta. && (10.2) \end{aligned}$$

Overall, this proves part (ii) of the claim of the lemma for round r.

It remains to show part (iii) of the claim, i.e., the bound on the skew of pulse $r+1$. To this end, fix any $v, w \in V_g$ and assume w.l.o.g. that $p_{v,r+1} \geq p_{w,r+1}$. We apply the second statement of Corollary 10.6, yielding

$$\begin{aligned} &p_{v,r+1} - p_{w,r+1} \\ &= H_v^{-1}(H_v(p_{v,r+1})) - H_w^{-1}(H_w(p_{w,r+1})) \\ &= H_v^{-1}(H_v(p_{v,r}) + \Delta_v(\tau_{v,r}) + T) - H_w^{-1}(H_w(p_{w,r}) + \Delta_w(\tau_{w,r}) + T) && \text{Line 13} \\ &\leq p_{v,r} + \Delta_v(\tau_{v,r}) + T - \left(p_{w,r} + \frac{\Delta_w(\tau_{w,r}) + T}{\vartheta}\right) && 1 \leq \tfrac{dH}{dt} \leq \vartheta \\ &= p_{v,r} + \Delta_v(\tau_{v,r}) - (p_{w,r} + \Delta_w(\tau_{w,r})) + \left(1 - \frac{1}{\vartheta}\right)(\Delta_w(\tau_{w,r}) + T) && \text{adding } 0 \\ &\leq \frac{\|\vec{p}_r\|}{2} + \delta + \left(1 - \frac{1}{\vartheta}\right)(\Delta_w(\tau_{w,r}) + T) && \text{Corollary 10.6} \\ &\leq \frac{\mathcal{S}}{2} + \delta + \left(1 - \frac{1}{\vartheta}\right)(\Delta_w(\tau_{w,r}) + T) && \text{prerequisite} \\ &\leq \frac{\mathcal{S}}{2} + \delta + \left(1 - \frac{1}{\vartheta}\right)(T + \mathcal{S} + \delta). && (10.2) \end{aligned}$$

In order for this upper bound to be at most $\mathcal{S}$, it is sufficient if

$$\left(1 - \frac{\vartheta}{2}\right)\mathcal{S} \geq (2\vartheta - 1)\delta + (\vartheta - 1)T.$$

This is equivalent to the prerequisite constraint on $\mathcal{S}$. This shows part (iii) of the claim for round r, completing the proof. □

Before we can prove our main theorem, we need to get a hold on δ. This is a straightforward calculation.

Lemma 10.8. *Suppose round r is executed correctly, $\|\vec{p}_r\| \leq \mathcal{S}$, and $v \in V_g$ receives the message from $w \in V_g$ for this round at time t. Then setting*

$$\Delta(w) := H_v(t) - H_v(p_{v,r}) - d + u - \mathcal{S}$$

is a suitable choice for the computation in Line 9, achieving $\delta \leq u + (\vartheta - 1)d + (\vartheta^2 + \vartheta - 2)\mathcal{S}$.

Proof. Denote by t_s the time when w sent the message that v receives at time t. We have that

$$\begin{aligned}
\tfrac{dH_v}{dt} \geq 1 \qquad & H_v(t) - H_v(p_{v,r}) \geq t - p_{v,r} \\
\text{adding } 0 \qquad & = t - t_s + t_s - p_{w,r} + p_{w,r} - p_{v,r} \\
\text{minimum delay} \qquad & \geq d - u + t_s - p_{w,r} + p_{w,r} - p_{v,r} \\
\tfrac{dH_w}{dt} \leq \vartheta \qquad & \geq d - u + \frac{H_w(t_s) - H_w(p_{w,r})}{\vartheta} + p_{w,r} - p_{v,r} \\
\text{Line } 5 \qquad & = d - u + \mathcal{S} + p_{w,r} - p_{v,r}.
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\tfrac{dH_v}{dt} \leq \vartheta \qquad & H_v(t) - H_v(p_{v,r}) \leq \vartheta(t - p_{v,r}) \\
\text{adding } 0 \qquad & = \vartheta(t - t_s + t_s - p_{w,r} + p_{w,r} - p_{v,r}) \\
\text{maximum delay} \qquad & \leq \vartheta(d + t_s - p_{w,r} + p_{w,r} - p_{v,r}) \\
\tfrac{dH_w}{dt} \geq 1 \qquad & \leq \vartheta(d + H_w(t_s) - H_w(p_{w,r} + p_{w,r} - p_{v,r}) \\
\text{Line } 5 \qquad & = \vartheta(d + \vartheta\mathcal{S} + p_{w,r} - p_{v,r}) \\
\text{adding } 0 \qquad & = \vartheta d + \vartheta^2\mathcal{S} + (\vartheta - 1)(p_{w,r} - p_{v,r}) + p_{w,r} - p_{v,r} \\
\text{prerequisite} \qquad & = \vartheta d + (\vartheta^2 + \vartheta - 1)\mathcal{S} + p_{w,r} - p_{v,r}.
\end{aligned}$$

Thus, $\Delta(w) = H_v(t) - H_v(p_{v,r}) - d + u - \mathcal{S}$ satisfies that

$$p_{w,r} - p_{v,r} \leq \Delta(w) \leq p_{w,r} - p_{v,r} + u + (\vartheta - 1)d + (\vartheta^2 + \vartheta - 2)\mathcal{S},$$

as claimed. □

Theorem 10.9. *Assume that $6 - 2\vartheta - \vartheta^2 - 2\vartheta^3 > 0$ and that estimates are computed according to Lemma 10.8. For any choice of*

$$T \geq \frac{(4\vartheta^3 + 2\vartheta^2 + 2\vartheta - 2)u + (4\vartheta^4 - 3\vartheta^3 - 2\vartheta^2 + 2)d}{6 - 2\vartheta - \vartheta^2 - 2\vartheta^3} \in O(d),$$

set

$$\mathcal{S} := \frac{2(2\vartheta - 1)(u + (\vartheta - 1)d) + 2(\vartheta - 1)T}{4 - 2\vartheta - \vartheta^2} \in O\left(u + \left(1 - \frac{1}{\vartheta}\right)T\right).$$

If $\max_{v \in V}\{H_v(0)\} \leq \mathcal{S}$*, then Algorithm 13 solves pulse synchronization with skew at most* $\mathcal{S}$*,* $P_{\min} \geq (T - (\vartheta + 1)\mathcal{S})/\vartheta$*, and* $P_{\max} \leq T + 3\mathcal{S}$*.*

Proof. Set $\delta := u + (\vartheta - 1)d + (\vartheta^2 + \vartheta - 2)\mathcal{S}$ in accordance with Lemma 10.8. Thus,

$$\begin{aligned} \mathcal{S} &= \frac{2(2\vartheta - 1)(u + (\vartheta - 1)d) + 2(\vartheta - 1)T}{4 - 2\vartheta - \vartheta^2} \\ &= \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta} > \delta. \end{aligned} \tag{10.3}$$

Moreover,

$$\begin{aligned} T &\geq \frac{(4\vartheta^3 + 2\vartheta^2 + 2\vartheta - 2)u + (4\vartheta^4 - 3\vartheta^3 - 2\vartheta^2 + 2)d}{6 - 2\vartheta - \vartheta^2 - 2\vartheta^3} \\ &= (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d. \end{aligned}$$

The claim is now shown by a straightforward induction on the pulse number, where the hypothesis includes that all previous rounds have been executed correctly. The induction is anchored at the first pulse, which satisfies the skew bounds due to the assumed bound on the hardware clock values at time 0. The induction step is performed by invoking Lemma 10.7, where Lemma 10.8 shows that δ is indeed a bound on the quality of estimates. We obtain that $\mathcal{S}$ is a bound on the skew for all pulses and that $(T - \mathcal{S})/\vartheta \leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} \leq T + \mathcal{S} + \delta$ for each $r \in \mathbb{N}_{>0}$. This implies for each $r \in \mathbb{N}_{>0}$ that

$$\begin{aligned} \min_{v \in V_g}\{p_{v,r+1}\} - \max_{v \in V_g}\{p_{v,r}\} &\geq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} + \mathcal{S} \\ &\geq \frac{T - (\vartheta + 1)\mathcal{S}}{\vartheta} \end{aligned}$$

and that

$$\begin{aligned} \max_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} &\leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} + \mathcal{S} \\ &\leq T + 2\mathcal{S} + \delta \\ &\overset{(10.3)}{<} T + 3\mathcal{S}. \end{aligned}$$

□

Remark 10.10.

1. *The theorem requires that* $6 - 2\vartheta - \vartheta^2 - 2\vartheta^3 > 0$*, which is the case for* $\vartheta \leq 1.09$*. As* ϑ *approaches this threshold, the skew goes to* ∞*.*

2. *Sending* $(T, \vartheta) \to (\infty, 1)$, *the ratio* $P_{\max}/P_{\min} \in (1 + o(1))\vartheta$. *However, when sending* $T \to \infty$ *while keeping* ϑ *fixed, the ratio converges to a constant* $c \in 1 + O(\vartheta - 1)$.
3. *If on initialization such a tight skew bound cannot be guaranteed, one can choose T accordingly larger.*
4. *Alternatively, one can only initially use the larger T and keep reducing T alongside the decrease in (the worst-case bound on) the skew.*
5. *A known bound on the initial skew is necessary for executing the algorithm.*
6. *We haven't clarified how nodes compute their estimates of faulty nodes' clocks. What if these nodes send no or many messages during a round? The answer is simple: It doesn't matter. As the approximate agreement algorithm works regardless of what values faulty nodes provide, choosing* any *default value for nodes clearly not obeying the protocol will do.*

Bibliography

[1] Boksberger, Philipp, Fabian Kuhn, and Roger Wattenhofer. 203. On the Approximation of the Minimum Maximum Stretch Tree Problem, Technical Report 409, ETH Zurich.

[2] Fisher, Allan L., and Hsiang-Tsung Kung. 1985. Synchronizing Large VLSI Processor Arrays. *IEEE Transactions on Computers* C-34 (8): 734–740.

[3] Hopkins, A. L., T. B. Smith, and J. H. Lala. 1978. Ftmp – a highly reliable fault-tolerant multiprocess for aircraft. *Proceedings of the IEEE* 66 (10): 1221–1239. doi:10.1109/PROC.1978.11113.

[4] Kopetz, H. 2003. Fault containment and error detection in the time-triggered architecture. In *The sixth international symposium on autonomous decentralized systems, 2003. isads 2003.*, 139–146. doi:10.1109/ISADS.2003.1193942.

[5] Pease, M., R. Shostak, and L. Lamport. 1980. Reaching agreement in the presence of faults. *J. ACM* 27 (2): 228–234. doi:10.1145/322186.322188. http://doi.acm.org/10.1145/322186.322188.

[6] Srikanth, T. K., and Sam Toueg. 1987. Optimal clock synchronization. *J. ACM* 34 (3): 626–645. doi:10.1145/28869.28876. https://doi.org/10.1145/28869.28876.