Exercise 5: Aligning our Clocks

Task 1: Converging to Agreement

a) Given a skew bound $\mathcal{S}_r$ for pulse r, determine T_r and δ_r so that performing a respective iteration of the loop of Algorithm 5.2 results in correct execution of round r.

b) Determine a skew bound $\mathcal{S}_{r+1}$ for pulse $r+1$ as function of $\mathcal{S}_r$ for the (minimal) choices of T_r and δ_r from a). What is $\mathcal{S}_\infty \coloneqq \lim_{r\to\infty} \mathcal{S}_r$?

c) Assume that $\max_{v\in V_g}\{H_v(0)\} \leq H$ for some known $H > \mathcal{S}_\infty$. Given ε, determine the round r_ε so that $\mathcal{S}_r \leq \mathcal{S}_\infty + \varepsilon$ for all $r \geq r_\varepsilon$. How long does it take in terms of real time until this skew bound is reached? (Hint: an asymptotic bound suffices, where we consider ϑ (and thus all values depending only on ϑ) to be a constant.)

d) Is this bound good/bad/optimal?

Solution

a) Following the lecture notes, we can directly use the lower bound on T from Lemma 5.8 and choose δ_r as in Lemma 5.9, yielding $T_r/\vartheta \geq (\vartheta^2+\vartheta+1)\mathcal{S}_r + \vartheta d$ and $\delta_r \coloneqq u + (\vartheta-1)d + 2(\vartheta^2-\vartheta)\mathcal{S}_r$.

b) Choosing δ_r and minimal T_r according to a), be can bound $\|\vec{p}_{r+1}\|$ analogously to the proof of Lemma 5.8, yielding

$$\begin{aligned}\|\vec{p}_{r+1}\| &\leq \frac{\mathcal{S}_r}{2} + \delta_r + \left(1-\frac{1}{\vartheta}\right)T_r \\ &= \frac{\mathcal{S}_r}{2} + u + (\vartheta-1)d + 2(\vartheta^2-\vartheta)\mathcal{S}_r + (\vartheta^3-1)\mathcal{S}_r + (\vartheta^2-\vartheta)d \\ &= \frac{(2\vartheta^3+4\vartheta^2-4\vartheta-1)\mathcal{S}_r}{2} + u + (\vartheta^2-1)d =: \mathcal{S}_{r+1}\,.\end{aligned}$$

If $q \coloneqq (2\vartheta^3+4\vartheta^2-4\vartheta-1)/2 < 1$ (which is equivalent to the requirement on ϑ from Theorem 5.10), we thus have

$$\lim_{r\to\infty}\mathcal{S}_r = \frac{u+(\vartheta^2-1)d}{1-q} = \frac{2u+2(\vartheta^2-1)d}{3+4\vartheta-4\vartheta^2-2\vartheta^3}\,.$$

c) We have that

$$\begin{aligned}\mathcal{S}_r &= q^{r-1}\mathcal{S}_1 + \sum_{k=0}^{r-2} q^k(u+(\vartheta^2-1)d) \\ &= q^{r-1}\mathcal{S}_1 + \frac{1-q^{r-1}}{1-q}\cdot(u+(\vartheta^2-1)d) \\ &= q^{r-1}H + (1-q^{r-1})\mathcal{S}_\infty\,.\end{aligned}$$

Hence, $\mathcal{S}_r \leq \mathcal{S}_\infty + \varepsilon$ is equivalent to

$$\begin{aligned} & q^{r-1}(H-\mathcal{S}_\infty) \leq \varepsilon \\ \Leftrightarrow \quad & \frac{H-\mathcal{S}_\infty}{\varepsilon} \leq \left(\frac{1}{q}\right)^{r-1} \\ \Leftrightarrow \quad & 1+\log_{1/q}\frac{H-\mathcal{S}_\infty}{\varepsilon} \leq r\end{aligned}$$

Thus, $r_\varepsilon \coloneqq 1 + \lceil \log_{1/q}(H - \mathcal{S}_\infty)(\varepsilon) \rceil$ is sufficient. In order to bound the real time until this bound is achieved, consider first the total nominal duration of all respective rounds:

$$\begin{aligned}
\sum_{r=1}^{r_\varepsilon - 1} T_r &= \sum_{r=1}^{r_\varepsilon - 1} \vartheta((\vartheta^2 + \vartheta + 1)\mathcal{S}_r + \vartheta d) \\
&= \sum_{r=1}^{r_\varepsilon - 1} \vartheta((\vartheta^2 + \vartheta + 1)(q^{r-1}H + (1 - q^{r-1})\mathcal{S}_\infty) + \vartheta d) \\
&< 3\vartheta^3 \sum_{r=1}^{r_\varepsilon - 1} ((q^{r-1}H + (1 - q^{r-1})\mathcal{S}_\infty) + d) \\
&< 3\vartheta^3 \left(\sum_{r=0}^{\infty} q^r (H - \mathcal{S}_\infty) + \sum_{r=1}^{r_\varepsilon - 1} (\mathcal{S}_\infty + d) \right) \\
&= 3\vartheta^3 \left(\frac{1}{1-q} \cdot (H - \mathcal{S}_\infty) + (r_\varepsilon - 1)(\mathcal{S}_\infty + d) \right) \\
&\in \mathcal{O}\left(H + (\mathcal{S}_\infty + d) \log_{1/q} \frac{H - \mathcal{S}_\infty}{\varepsilon} \right) \\
&= \mathcal{O}\left(H + d \log_{1/q} \frac{H - \mathcal{S}_\infty}{\varepsilon} \right)
\end{aligned}$$

Here, the second to last step assumes that $H > \mathcal{S}_\infty + \varepsilon$ (i.e., $r_\varepsilon \geq 2$), while the last step uses that $\mathcal{S}_\infty \in \mathcal{O}(u + (\vartheta - 1)d) \subseteq \mathcal{O}(d)$. While hardware clock rates are at least 1, it is possible that clocks are set back, resulting in rounds that are longer than time T_r. However, we know from the analysis that the maximum duration is $T_r + \delta_r$, where $\delta_r \leq T_r$, so the asymptotic bound does not change.

d) The first summand of H is clearly necessary, as this is the time required for the correct nodes to even generate the first pulse. The factor of d in the second summand is also clearly necessary, as some communication must occur in order to synchronize the nodes. Thus, the only possible overhead is the logarithmic factor due to the exponential convergence towards $\mathcal{S}_\infty$. This factor is inherent to the technique (i.e., approximate agreement), but not known to be necessary for clock synchronization in the presence of faults.

Task 2: We're not Synchronized!

a) Fix any T and $\mathcal{S}$ in accordance with Theorem 5.10, and compute Δ_w^v as in Lemma 5.9. Assume that node v uses default value 0 for Δ_w^v if no (or conflicting) messages are received from w during a round. Under these conditions, give an execution of Algorithm 5.2 in which skews remain larger than $T/2$ forever. You may assume that ϑ is sufficiently small to simplify matters, and negative hardware clock values are permitted (these represent late initialization).

b) Now assume that there are $n - f \leq n - 2$ correct nodes $v \in V_g$ satisfying $0 \leq H_v(0) \leq \mathcal{S}$ and you are given an execution in which the skew is $\mathcal{S}$ for each pulse, and each correct node generates a pulse exactly every $P \in \mathbb{R}^+$ time. Moreover, faulty nodes never send messages and you may assume that the algorithm's parameters are such that, potentially, Δ_w^v could become much larger than $\mathcal{S}$ (in a correct execution of the algorithm). Show that if one of the faulty nodes is merely a "confused" correct node whose initial hardware clock value is off, there is an execution in which this node never synchronizes with the others. (Hint: Don't crunch numbers, find a way

of giving the faulty nodes control over the confused node's clock adjustments, and use this to keep it away from the others!)

c*) Can you fix this by modifying the algorithm? That is, make sure that in the scenario of b), but even with up to $f-1 < n/3$ Byzantine nodes, eventually all correct nodes have skew at most $\mathcal{S}$? Again, you may assume that ϑ is close to 1. (Hint: Modify the behavior of nodes when they have *proof* that something is amiss so that they either catch up with the main field or slow down enough for the main field to catch up with them.)

Solution

a) We split the nodes into two groups $A \dot\cup B$ so that $\min\{|A|, |B|\} > f$. Nodes $v \in A$ have $H_v(0) = 0$, nodes in $v \in B$ have $H_v(0) = -H := -(T - \vartheta^2\mathcal{S} - \vartheta d - u/2)$. We choose $\vartheta > 1$ sufficiently small such that $H > \vartheta^2\mathcal{S} + \vartheta d$; as with $\vartheta = 1$ we had that $T \geq 6(u+d)$ and $\mathcal{S} = 2u$ and the relevant expressions are for $\vartheta \geq 1$ continuous functions of ϑ, this is possible. Note also that for sufficiently small $\vartheta > 1$, we have that $H \geq T/2$. All hardware clock rates are 1 and all message delays are $d - u/2$. Due to symmetry, all nodes in A will have the same logical clock values at all times, and the same holds true for the nodes in B.

Suppose $w \in A$ ($w \in B$) sends a message at time t, i.e., $L_v(t) = (r-1)T + (\vartheta+1)\mathcal{S}$. This message is received at time $t + d - u/2$, when $v \in A$ ($v \in B$) has

$$L_v\left(t + d - \frac{u}{2}\right) = L_w(t) + d - \frac{u}{2} = (r-1)T + (\vartheta+1)\mathcal{S} + d - \frac{u}{2}$$

for some $r \in \mathbb{N}$, as $d - u/2 < \mathcal{S} + \vartheta d$, i.e., v has not adjusted its clock during this iteration of the loop of the algorithm yet. Thus, if v receives no other messages during $[p_{v,r}, \tau_{v,r}]$, it computes $S_v^{(f+1)} = -(\vartheta-1)d - u/2 - (\vartheta^2 - \vartheta)\mathcal{S}$ and $S_v^{(n-f)} = 0$, setting its logical clock back by $\lambda := ((\vartheta-1)d + u/2 + (\vartheta^2-\vartheta)\mathcal{S})/2$ at time $\tau_{v,r}$. This results in iteration r of the loop being complete after $T + \lambda$ real time.

We claim that this happens every time at all nodes. To show this, assume for contradiction that this is not true, and let r be the minimal iteration number in which this occurs. Assume first that a node $v \in A$ receives a message from $w \in B$ during $[p_{v,r}, \tau_{v,r}]$, which w sent at time $t \leq \tau_{v,r} - d + u$. As the claim was not violated before, we have that

$$\begin{aligned} p_{v,r} &= (r-1)(T+\lambda) + \mathcal{S} \\ \tau_{v,r} &= (r-1)(T+\lambda) + (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d \\ t &= (r'-1)(T+\lambda) + (\vartheta+1)\mathcal{S} + H \end{aligned}$$

for some $r' \in \mathbb{N}$. As $t \in [p_r - d + u/2, \tau_{v,r} - d + u/2]$, it follows that

$$0 \leq (r'-r)(T+\lambda) + \vartheta\mathcal{S} + d - \frac{u}{2} + H \leq (\vartheta^2+\vartheta)\mathcal{S} + \vartheta d\,.$$

As $H > \vartheta^2\mathcal{S} + \vartheta d$, the upper bound enforces $r' < r$. However, $T > \vartheta\mathcal{S} + d + H$, violating the lower bound. Hence, consider the other case in which $v \in B$ receives a message from $w \in A$ during $[p_{v,r}, \tau_{v,r}]$ instead. As the claim was not violated before, we have that

$$\begin{aligned} p_{v,r} &= (r-1)(T+\lambda) + \mathcal{S} + H \\ \tau_{v,r} &= (r-1)(T+\lambda) + (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d + H \\ t &= (r'-1)(T+\lambda) + (\vartheta+1)\mathcal{S} \end{aligned}$$

yielding that

$$0 \leq (r'-r)(T+\lambda) + \vartheta\mathcal{S} + d - \frac{u}{2} - H \leq (\vartheta^2+\vartheta)\mathcal{S} + \vartheta d\,.$$

As $H > \vartheta\mathcal{S} + d$, it must hold that $r' > r$. However, the lower bound then implies that

$$T < H + \vartheta^2\mathcal{S} + (\vartheta - 1)d + \frac{u}{2}\,,$$

which is not the case. We conclude that the claim holds for all times. In particular, the skew between nodes in $v \in B$ and $v \in A$ is

$$p_{v,r} - p_{w,r} = H \geq \frac{T}{2}$$

for all $r \in \mathbb{N}$.

b) Observe that the confused node v would produce pulses exactly every T time if it never received any messages during $[p_{v,r}, \tau_{v,r}]$ and its hardware clock rate is 1. From Theorem 5.10, we know that $P \in [T/\vartheta - \mathcal{S}, T + 2\mathcal{S}]$, where $(1 - 1/\vartheta)T < \mathcal{S}$. Accordingly, $P \in T \pm \mathcal{O}(\mathcal{S})$.

If $T < P$, choose the initial hardware clock value of v such that it misses all but f messages from correct nodes in each pulse, which it receives very early (i.e., during $[p_{v,r}, p_{v,r} + \mathcal{S}]$). Accordingly, v will compute values $\Delta_w^v \ll -\mathcal{S}$ for these nodes. Thus, if a faulty node w now sends a message such that the resulting $\Delta_w^v \in [-\mathcal{O}(\mathcal{S}), 0]$, v will set its clock forward by $\Delta_w^v/2$ (as it defaults to $\Delta_w^v = 0$ for the missing messages). Recalling that Δ_w^v can be large compared to $2\mathcal{S}$, the faulty nodes can thus indeed make sure that v produces pulses exactly every P time and this scenario repeats.

Similarly, if $T \geq P$, choose the initial hardware clock value of v such that it misses all but f messages from correct nodes in each pulse, which it receives very late (i.e., during $[\tau_{v,r} - \mathcal{S}, \tau_{v,r}]$). Then faulty nodes can make v set its clock back sufficiently to match the period P or the other nodes and stay in this state indefinitely.

c*) See https://people.mpi-inf.mpg.de/ clenzen/pubs/KL18self-stabilizing.pdf .