8 Gradient Clock Synchronization

Chapter Contents

Learning Goals

todo

8.1 Overview

In Chapter 7, we have studied the clock synchronization problem. We have see that a global skew of $\Theta(uD + (\vartheta - 1)d)$ is worst-case optimal, where D is the diameter of the network graph (with crashed nodes removed). However, if the logical clocks are intended to clock the system, the global skew is not determining the clock frequency the system can sustain under synchronous operation. Rather, the essential property is the clock skew between nodes (i.e., clock domains) that communicate with each other.

In this chapter, we will assume that the network graph corresponds to the communication graph, i.e., if the clock domains corresponding to nodes $v \in V$ and $w \in V$ communicate directly, then there is also an edge $\{v, w\} \in E$ in the network graph $G = (V, E)$ on which we solve the clock synchronization problem. In this setting, a highly relevant quality measure for synchronization is the *local skew.*

Definition 8.1 (Local Skew)**.** *Given an algorithm that computes logical clocks $L_v(t)$, $t \in \mathbb{R}_{\geq 0}$, at each node $v \in V$, define its* local skew *as*

$$\mathcal{L} := \sup_{t \in \mathbb{R}_{\geq 0}} \{\mathcal{L}(t)\},$$

over all executions $\mathcal{E}$, where

$$\mathcal{L}(t) := \max_{\{v,w\} \in E} \{|L_v(t) - L_w(t)|\}.$$

We study the local skew for clock synchronization algorithms in TMP.

One might hope that the local skew can be kept much smaller than the global skew. In fact, since the lower bound on the global skew given in Theorem 7.12 is based on “hiding” a large clock skew between nodes in distance D from each other, one might venture the guess that a local skew of $O(u + (\vartheta - 1)d)$ can be guarenteed. Our first main result in this chapter shows that such an ideal distribution of the global skew over the network cannot always be maintained with bounded logical clock rates.

Theorem 8.4. *Any clock synchronization algorithm satisfying that*

$$\frac{dH_v}{dt}(t) \leq \frac{dL_v}{dt}(t) \leq (1 + \mu)\frac{dH_v}{dt}(t)$$

for all nodes v and times t has

$$\mathcal{L} \geq \left(\frac{u}{4} - (\vartheta - 1)d\right) \log_{\lceil \sigma \rceil} D,$$

where $\sigma := \mu/(\vartheta - 1)$.

This lower bound constraints clock rates from below by $\frac{dH_v}{dt}(t) \geq 1$ and from above by $(1+\mu)\frac{dH_v}{dt}(t) \leq \vartheta(1+\mu)$. Recall that the lower bound is motivated by requiring that clocks make progress; however, as can be seen by analyzing a variant of Algorithm 2 in TMP, allowing for an *amortized* logical clock rate[10] of at least 1 enables us to keep the local skew constant.

Theorem 8.8. *There is a clock synchronization algorithm achieving a local skew of* $\max\{d, \vartheta u\}$*, amortized* 1*-progress with* $C = 0$*, and* $\frac{dL_v}{dt}(t) \leq \frac{dH_v}{dt}(t)$ *for all times* t *and nodes* $v \in V$*.*

The algorithm providing these guarantees locally halts the logical clock for up to uD time. Although a formal statement would be more convoluted,[11] the proof of Theorem 8.4 reveals that, roughly speaking, this is necessary to ensure a local skew of $O(u)$.

But what if halting or dramatically slowing down the clocks is a problem? If the system is required to respond to local events as quickly as possible, one would want the the logical clocks that drive the nodes' computations are guaranteed to make progress at all times. In this setting, the requirement that $\frac{dL_v}{dt}(t) \geq \frac{dH_v}{dt}(t)$ should be upheld. So what about the upper bound, i.e., that $\frac{dL_v}{dt}(t) \leq (1+\mu)\frac{dH_v}{dt}(t)$? It turns out that choosing $\mu \geq \log_{1/(\vartheta-1)} D$ is of no use, as an algorithm utilizing faster clocks will inadvertantely introduce a large skew due to neighors' not being able to keep track of each others clocks any more.

Theorem 8.10. *Any clock synchronization algorithm satisfying that* $\frac{dH_v}{dt}(t) \leq \frac{dL_v}{dt}(t)$ *for all nodes* v *and times* t *has*

$$\mathcal{L} = \Omega\left(\left(\frac{u}{4} - (\vartheta - 1)d\right)\log_{\lceil\sigma\rceil} D\right)$$

for $\sigma = \log_{1/(\vartheta-1)} D)/(\vartheta - 1)$.

A painfully more elaborate argument shows that the same holds for $\sigma = \Theta(1/(\vartheta - 1))$ [**?**], but due to being executed in a different model, it does not immediately provide a corresponding statement in our setting. In favor of simplicity and intuition, we stick to the weaker bound.

Note that for $(1+\mu) \leq \vartheta$, it is impossible for logical clocks of nodes with $\frac{dH_v}{dt} = 1$ to catch up with the logical clocks of nodes with $\frac{dH_v}{dt} = \vartheta$. Thus, the above results lead to the question whether for the range of $\mu > \vartheta - 1$ the lower

[10] The theorem states amortized 1-progress for $C = 0$, but recall that for notational convenience we assume that all nodes wake up at time 0. If the last node wakes up at time t_0, then $C = t_0$.

[11] The clocks do not need to be halted, but they must progress sufficiently slow to prevent the build-up of skew the lower bound accomplishes.

bound from Theorem 8.4 can be (up to constants) matched by a corresponding algorithm. The second main result of this chapter is that this is indeed the case, provided that nodes can estimate the logical clock values of their neighbors up to an error of $\delta = O(u)$ at all times and $\mathcal{G}/\delta = O(D)$.

Theorem 8.28. *Suppose that $\kappa \geq \delta$ and $H_v(0) - H_w(0) \leq \kappa$ for all edges $\{v, w\} \in E$. Then Algorithm 7 maintains a local skew of*

$$\mathcal{L} \leq 2\kappa \left\lceil \log_\sigma \frac{\mathcal{G}}{\kappa} \right\rceil,$$

where $\sigma := \mu/(\vartheta - 1)$.

This means that, while we are not able to guarantee a local skew that is entirely independent of D, the dependence on D is only logarithmic. Moreover, note that the base of the logarithm can become very large, at the expense of a larger "drift" of logical clocks than of hardware clocks.

With the technical machinery established for bounding the local skew we can also bound the global skew.

Theorem 8.29. *Assume that $\kappa \geq \delta$ and $H_v(0) - H_w(0) \leq \kappa$ for all $\{v, w\} \in E$. Then Algorithm 7 satisfies $\mathcal{G} \leq (1 + 1/(\sigma - 1))\kappa D$, where $\sigma = \mu/(\vartheta - 1)$.*

We then proceed to show that $\delta = O(u)$ is easily achieved, provided that $\mu = O(u/d)$. Together, under the mild constraint that $u/4 - (\vartheta - 1)d = \Omega(u)$ this implies that Algorithm 7 is, up to constant factors, simultaneously optimal with respect to both local and global skew, for any choice of $\vartheta - 1 < \mu = O(u/d)$.

Corollary 8.32. *Suppose that*

- $H := \max_{\{v,w\} \in E}\{H_v(0) - H_w(0)\} \in O(u)$,
- *Algorithm 8 with $T = d$ is used to compute clock estimates and is initialized at all nodes by time $-2d$,*
- $2(\vartheta - 1) \leq \mu = O(u/d)$, *and*
- $\kappa = \max\{H, \delta\}$, *where δ is as in Lemma 8.31.*

Then Algorithm 7 guarantees that

- $\frac{dH_v}{dt}(t) \leq \frac{dL_v}{dt}(t) \leq (1 + \mu)\frac{dH_v}{dt}(t)$ *for all nodes v and times t,*
- $\mathcal{G} = O(D)$, *and*
- $\mathcal{L} = O(u \log_\sigma D)$, *where $\sigma = \mu/(\vartheta - 1)$.*

Remark 8.2. *The constraints on μ cannot be met if $(\vartheta - 1)d \ll u$. For the global skew, we could easily resolve this by constructing better hardware clocks using the more accurate wire delays. However, as messages are under way for* *up to d time, such clocks must suffer from an additive error term of* $(\vartheta - 1)d$. *Naturally, this is also a lower bound on the local skew and the quality of estimates of neighbors' clocks. Thus, it comes without surprise that this term cannot be easily removed from the local skew bound, which it enters via* δ, *which bounds* κ *from below.*

Remark 8.3. *By detecting crashes using timeouts in a similar manner as we used to derive Theorem 7.6, Algorithm 7 can handle crash faults without changing skew bounds in a relevant manner: if a node crashes, this can be recognized* $O(T + d)$ *time later by its neighbors due to missing clock updates. By keeping the clock of the crash node running fictitiously for additional* $O(d)$ *time, this delay can be incorporated into* δ *without increasing it significantly. Similarly, Theorem 8.8 can be adapted, where detecting crashes by timeouts only changes that we need to choose a non-zero* $C = O(dD)$. *Note that in both cases, at each time* D *then is to be replaced by the maximum diameter of the network up to time* t.

As assuming crash faults is too optimistic in low-level hardware implementations, we refrain from formalizing these results.

8.2 Lower Bound on the Local Skew with Bounded Clock Rates

In Chapter 7, we proved essentially matching upper and lower bounds on the worst-case global skew for the clock synchronization problem. We saw that during an execution of the Max algorithm (Algorithm 5), all logical clocks in all executions eventually agree up to an additive term of $O(uD)$ (ignoring other parameters). The lower bound we proved in Theorem 7.12 shows that a global skew of $\Omega(uD)$ is unavoidable for any algorithm in which clocks run at an amortized constant rate, at least in the worst case. In our lower bound construction, the two nodes v and w that achieved the maximal skew were distance D apart. However, the lower bound did not preclude neighboring nodes from remaining closely synchronized throughout an execution. As we will see in Theorem 8.8, this is indeed possible if one is willing to slow down clocks arbitrarily (or simply stop them), even if the amortized rate is constant.

We now look into what happens if one requires that clocks progress at a constant rate at all times. That is, we constrain logical clocks to increase at rates between $\frac{dH}{dt}$ and $(1 + \mu)\frac{dH}{dt}$ at all times.

Before proving Theorem 8.4, we provide some intuition. Assume that $(\vartheta - 1)d \ll u$, so we can ignore terms with $(\vartheta - 1)d$ for the moment and drop them from the notation. The basic strategy of the proof is to construct a sequence of executions $\mathcal{E}_0, \mathcal{E}_1, \dots, \mathcal{E}_\ell$ and times $t_0 < t_1 < \cdots < t_\ell$ such that at each time t_i, there exist nodes v_i, w_i satisfying $L_{v_i}(t_i) - L_{w_i}(t_i) \geq i\alpha u \cdot \text{dist}(v_i, w_i)$, for some suitable constant α. Our construction works up to $\ell = \Omega(\log_\sigma D)$ with $\mathrm{dist}(v_\ell, w_\ell) = 1$, which gives the desired result.

In more detail, the idea of the proof is to use the "shifting" technique of Lemma 7.11 applied ℓ times to closer and closer pairs of nodes. By Lemma 7.11, there is an execution $\mathcal{E}_0$ and a pair of nodes v_0, w_0 satisfying $\mathrm{dist}(v_0, w_0) = D$ such that at time $t_0 = d + (u/(2(\vartheta - 1)) - d)D$, we have $L_{v_0}(t_0) - L_{w_0}(t_0) \gtrapprox u/2 \cdot \mathrm{dist}(v_0, w_0)$. Fix a shortest path P from v_0 to w_0. For any pair of nodes v, w along P, we define the *average skew* between v and w at time t to be $|L_v(t) - L_w(t)|/\mathrm{dist}(v, w)$. In particular, the average skew between v_0 and w_0 is at least (roughly) $u/2$.

We extend the execution $\mathcal{E}_0$ for $t > t_0$ by setting all hardware clock rates to 1 for $t > t_0$ and all message delays to $d - u/2$ (as in the execution $\mathcal{E}$ in Lemma 7.11). Thus, by assumption at times $t > t_0$ logical clock rates are always between $\frac{dH}{dt} = 1$ and $(1+\mu)\frac{dH}{dt} = 1 + \mu$. Hence, for every $t > t_0$ in the extended execution, we have $L_{v_0}(t) - L_{w_0}(t) \geq u/2 \cdot \mathrm{dist}(v_0, w_0) - \mu \cdot (t - t_0)$. That is, the average skew between v_0 and w_0 decreases at a rate of at most $\mu/\mathrm{dist}(v_0, w_0)$. By taking

$$t_1 = t_0 + d + \left(\frac{u}{2} \cdot (\vartheta - 1) - d\right) \cdot k \approx t_0 + \frac{u}{2} \cdot (\vartheta - 1)$$

for some suitably chosen k, there exists a pair of nodes v_1, w_1 on P with $\mathrm{dist}(v_1, w_1) = k$ such that the average skew between v_1 and w_1 at time t_1 is (roughly) at least

$$\frac{u}{2} - \frac{\mu}{\mathrm{dist}(v_0, w_0)} \cdot (t_1 - t_0) = \frac{u}{2} - \frac{u}{2} \cdot \frac{\mu}{\vartheta - 1} \cdot \frac{k}{\mathrm{dist}(v_0, w_0)}$$

in the execution $\mathcal{E}_0$. Recalling that $\sigma = \mu/(\vartheta - 1)$ and choosing $k = \mathrm{dist}(v_0, w_0)/\lceil 2\sigma \rceil$, this is at least $u/4$. We then apply the shifting technique again to the nodes v_1 and w_1 on the interval $[t_0, t_1]$. In this way we define an execution $\mathcal{E}_1$ in which there is a time when the skew between v_1 and w_1 is by roughly $uk/2$ larger than the skew in $\mathcal{E}_0$ at time t_1. Therefore, in $\mathcal{E}_1$, the average skew bewtween v_1 and w_1 reaches about $3/4u$.

We then iterate the procedure above $\ell \lfloor \log_{\lceil 2\sigma \rceil} D \rfloor$ times. In the i-th iteration, we obtain a pair of nodes v_i, w_i at distance $D/\lfloor 2\sigma \rfloor^i$ such that the average skew between v_i and w_i is at least $(1/2 + i/4) \cdot u$. Thus, after ℓ iterations, the skew between adjacent nodes v_ℓ and w_ℓ is roughly $u/2 \cdot \log_\sigma D$, which gives the desired result.

Theorem 8.4. *Any clock synchronization algorithm satisfying that*

$$\frac{dH_v}{dt}(t) \leq \frac{dL_v}{dt}(t) \leq (1 + \mu)\frac{dH_v}{dt}(t)$$

for all nodes v and times t has

$$\mathcal{L} \geq \left(\frac{u}{4} - (\vartheta - 1)d\right) \log_{\lceil \sigma \rceil} D,$$

where $\sigma := \mu/(\vartheta - 1)$.

Proof. Note that the claim is vacuous if $(\vartheta - 1)d \geq u/4$, so we can assume the opposite in the following. Set $b := \lceil 2\sigma \rceil$ and $i_{\max} := \lfloor \log_b D \rfloor$. By induction over $i \in [i_{\max} + 1]$, we show that we can build up a skew of $(i+2)(u/4 - (\vartheta - 1)d)\,\mathrm{dist}(v, w)$ between nodes $v, w \in V$ in distance $\mathrm{dist}(v, w) = b^{i_{\max} - i}$ at a time t_i in execution $\mathcal{E}^{(i)}$, such that after time t_i all hardware clock rates are 1 and all sent messages have delays of $d - u/2$.

We anchor the induction at $i = 0$ by applying Lemma 7.11, choosing t_0 as in the lemma. We pick two nodes $v, w \in V$ in distance $b^{i_{\max}} \leq D$ of each other such that $L_v^{(\mathcal{E}_1)}(t_0) \geq L_w^{(\mathcal{E}_1)}(t_0)$. Now consider $\mathcal{E}_v$ for this choice of $v, w \in V$, which satisfies $H_v^{(\mathcal{E}_v)}(t_0) = H_v^{(\mathcal{E}_1)}(t_0) + (u/2 - (\vartheta - 1)d)\,\mathrm{dist}(v, w)$ and $H_w^{(\mathcal{E}_v)}(t_0) = H_w^{(\mathcal{E}_1)}(t_0)$. Denote by $t < t_0$ the time such that $H_v^{(\mathcal{E}_v)}(t) = H_v^{(\mathcal{E}_1)}(t_0)$. We get that

$$\begin{aligned}
&\text{adding 0} & L_v^{(\mathcal{E}_v)}(t_0) &= L_v^{(\mathcal{E}_v)}(t) + L_v^{(\mathcal{E}_v)}(t_0) - L_v^{(\mathcal{E}_v)}(t) \\
&\tfrac{dL_v}{dt} \geq \tfrac{dH_v}{dt} & &\geq L_v^{(\mathcal{E}_v)}(t) + H_v^{(\mathcal{E}_v)}(t_0) - H_v^{(\mathcal{E}_v)}(t) \\
&\text{by definition} & &= L_v^{(\mathcal{E}_v)}(t) + \left(\frac{u}{2} - (\vartheta - 1)d\right) \mathrm{dist}(v, w) \\
&\text{by indist.} & &= L_v^{(\mathcal{E}_1)}(t_0) + \left(\frac{u}{2} - (\vartheta - 1)d\right) \mathrm{dist}(v, w). \qquad (8.1)
\end{aligned}$$

We conclude that

$$\begin{aligned}
&\text{by indist.} & L_v^{(\mathcal{E}_v)}(t_0) - L_w^{(\mathcal{E}_v)}(t_0) &= L_v^{(\mathcal{E}_v)}(t_0) - L_w^{(\mathcal{E}_1)}(t_0) \\
&(8.1) & &\geq L_v^{(\mathcal{E}_1)}(t_0) + \left(\frac{u}{2} - (\vartheta - 1)d\right) \mathrm{dist}(v, w) - L_w^{(\mathcal{E}_1)}(t_0) \qquad (8.2) \\
&\text{choice of } v, w & &\geq \left(\frac{u}{2} - (\vartheta - 1)d\right) \mathrm{dist}(v, w).
\end{aligned}$$

We obtain $\mathcal{E}^{(0)}$ by changing all hardware clock rates in $\mathcal{E}_v$ to 1 at time t_0 and all message delays of messages sent at or after time t_0 to $d - u/2$. As this does not affect the logical clock values at time t_0—$\mathcal{E}^{(0)}$ is indistinguishable from $\mathcal{E}_v$ at $x \in V$ until local time $H_x^{(\mathcal{E}^{(0)})}(t_0)$—this shows the claim for $i = 0$.

For the induction step from i to $i + 1$, let $v, w \in V$, $\mathcal{E}^{(i)}$, and t_i be given by the induction hypothesis, i.e.,

$$L_v^{(\mathcal{E}^{(i)})}(t_i) - L_w^{(\mathcal{E}^{(i)})}(t_i) \geq (i+2)\left(\frac{u}{4} - (\vartheta - 1)d\right) \mathrm{dist}(v, w)\,,$$

and from time t_i on all hardware clock rates are 1 and sent messages have delay $d - u/2$. Note that the latter conditions mean that $\mathcal{E}^{(i)}$ behaves exactly like $\mathcal{E}_1$ from Lemma 7.11 from time t_i on, except that some messages sent at times $t < t_i$ may arrive during $[t_i, t_i + d)$. Hence, if we apply the same modifications to $\mathcal{E}^{(i)}$ as to $\mathcal{E}_1$, but starting from time $t_i + d$ instead of time 0, analogously to the lemma we show the following. For any $v', w' \in V$, we can construct an execution $\mathcal{E}_{v'}$ indistinguishable from $\mathcal{E}^{(i)}$, such that

- for all $x \in V$ and $t \geq t_i$, $H_x^{(\mathcal{E}^{(i)})}(t) = H_x^{(\mathcal{E}^{(i)})}(t_i) + t - t_i$,
- $H_{v'}^{(\mathcal{E}_{v'})}(t) = H_{v'}^{(\mathcal{E}^{(i)})}(t) + \operatorname{dist}(v', w')(u/2 - (\vartheta - 1)d)$ for all times $t \geq t_i + d + (u/(2(\vartheta - 1)) - d)\operatorname{dist}(v', w')$, and
- $H_{w'}^{(\mathcal{E}_{v'})}(t) = H_{w'}^{(\mathcal{E}^{(i)})}(t_i) + t - t_i$ for all $t \geq t_i$.

Consider the logical clock values of v and w in $\mathcal{E}^{(i)}$ at time

$$t_{i+1} := t_i + d + \left(\frac{u}{2(\vartheta - 1)} - d\right)\frac{\operatorname{dist}(v, w)}{b}.$$

Recall that $\frac{dL_v}{dt}(t) \geq \frac{dH_v}{dt}(t) \geq 1$ and $\frac{dL_w}{dt}(t) \leq (1 + \mu)\frac{dH_w}{dt}(t)$ at all times t. As $\frac{dH_w^{(\mathcal{E}^{(i)})}}{dt}(t) = 1$ at times $t \geq t_i$, we obtain

$$L_v^{(\mathcal{E}^{(i)})}(t_{i+1}) - L_w^{(\mathcal{E}^{(i)})}(t_{i+1}) \geq L_v^{(\mathcal{E}^{(i)})}(t_i) - L_w^{(\mathcal{E}^{(i)})}(t_i) - \mu(t_{i+1} - t_i)\,. \quad (8.3)$$

Recall that $\operatorname{dist}(v, w) = b^{i_{\max} - i}$ and that $b = \lceil 2\sigma \rceil$. We split up a shortest path from v to w in b subpaths of length $b^{i_{\max} - (i+1)}$. By the pidgeon hole principle, at least one of these paths must exhibit at least a $1/b$ fraction of the skew between v and w, i.e., there are $v', w' \in V$ with $\operatorname{dist}(v', w') = b^{i_{\max} - (i+1)} = \operatorname{dist}(v, w)/b$

so that

$$
\begin{aligned}
& && L_{v'}^{(\mathcal{E}^{(i)})}(t_{i+1}) - L_{w'}^{(\mathcal{E}^{(i)})}(t_{i+1}) \\
&\text{pidgeon hole} && \geq \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i+1}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i+1})}{b} \\
&\text{(8.3)} && \geq \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i}) - \mu(t_{i+1}-t_i)}{b} \\
&\text{by definition} && = \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i}) - \mu(d + (u/(2(\vartheta-1)) - d)\operatorname{dist}(v',w'))}{b} \\
&\operatorname{dist}(v',w') \geq 1 && \geq \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i}) - \mu u \operatorname{dist}(v',w')/(2(\vartheta-1))}{b} \\
&b = \lceil 2\sigma \rceil && \geq \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i})}{b} - \frac{\mu}{2\sigma(\vartheta-1)} \cdot \frac{u}{2} \cdot \operatorname{dist}(v',w') \\
&\sigma = \mu/(\vartheta-1) && = \frac{L_{v}^{(\mathcal{E}^{(i)})}(t_{i}) - L_{w}^{(\mathcal{E}^{(i)})}(t_{i})}{b} - \frac{u}{4} \cdot \operatorname{dist}(v',w') \\
&\text{induction hyp.} && \geq \frac{(i+2)(u/4-(\vartheta-1)d)\operatorname{dist}(v,w)}{b} - \frac{u}{4} \cdot \operatorname{dist}(v',w') \\
&\operatorname{dist}(v',w') = \operatorname{dist}(v,w)/b && = \left((i+2)\left(\frac{u}{4} - (\vartheta-1)d\right) - \frac{u}{4}\right)\operatorname{dist}(v',w'). && \text{(8.4)}
\end{aligned}
$$

In other words, as the average skew on a shortest path from v to w did not decrease by more than $u/4$, there must be a subpath of length $\operatorname{dist}(v,w)/b$ with at least the same average skew. Now we sneak in additional skew by advancing the (hardware and thus also logical) clock of v' using the indistinguishable execution $\mathcal{E}_{v'}$. By an analogous derivation to that of (8.2), we get that

$$
\begin{aligned}
& && L_{v'}^{(\mathcal{E}_v)}(t_{i+1}) - L_{w'}^{(\mathcal{E}_v)}(t_{i+1}) \\
&\text{anal. to (8.2)} && \geq L_{v'}^{(\mathcal{E}^{(i)})}(t_{i+1}) + \left(\frac{u}{2} - (\vartheta-1)d\right)\operatorname{dist}(v',w') - L_{w'}^{(\mathcal{E}^{(i)})}(t_{i+1}) \\
&\text{(8.4)} && \geq (i+3)\left(\frac{u}{4} - (\vartheta-1)d\right)\operatorname{dist}(v',w').
\end{aligned}
$$

This completes the induction. Plugging in $i = i_{\max}$ and noting that $\log b = \log\lceil 2\sigma \rceil \leq 1 + \log\lceil\sigma\rceil$, we get an execution in which two nodes at distance $b^0 = 1$ exhibit a skew of at least

$$
\begin{aligned}
&i_{\max} = \lfloor \log_b D \rfloor && (i_{\max}+2)\left(\frac{u}{4} - (\vartheta-1)d\right) \geq \left(\frac{u}{4} - (\vartheta-1)d\right)(1 + \log_b D) \\
& && \geq \left(\frac{u}{4} - (\vartheta-1)d\right)\log_{\lceil\sigma\rceil} D\,. \qquad \square
\end{aligned}
$$

- It is somewhat "bad form" to adapt Lemma 7.11 on the fly, as we did in the proof. However, the alternative of carefully defining partial executions, how to stitch them together, and proving indistinguishability results in this setting would mean to crack a nut with a sledgehammer.

- By making the base of the logarithm larger (i.e., making paths shorter more quickly), we can reduce the "loss" of skew in each step. Thus, we get a skew of $u/2 - (\vartheta - 1)d - \varepsilon$ per iteration, at the cost of reducing the number of iterations by a factor of $\log \sigma / (\log \sigma - \log \varepsilon^{-1})$. As typically $\sigma \gg 1$, this means that we gain roughly a factor of 2.
- We can gain another factor of 2 by introducing skew more carefully. If we construct $\mathcal{E}_1$ so that messages "in direction of w" have delay (roughly) $d - u$ and messages "in direction of v" have delay d, we can hide u skew per hop. We favored the simpler construction to avoid additional bookkeeping.
- Overall, if $(\vartheta - 1)d \ll u$, $\sigma \gg 1$, and $\log_\sigma D \gg 1$, we can show a lower bound of $(u - \varepsilon) \log_\sigma D$ for some small $\varepsilon > 0$.
- What if $(\vartheta - 1)d$ is comparable to u or even larger? As for a lower bound construction we can always pretend that clock drifts are actually smaller, e.g., $\vartheta' := \min\{\vartheta, 1 + u/(4d)\}$, the lower bound does not get weaker if the hardware clocks get worse. On the other hand, we will see that larger ϑ is not really an issue (up to a "one-time" additive term of $O((\vartheta - 1)d)$), as we can then bounce messages back and forth between nodes to keep track of time with greater accuracy than the "base clocks" permit.

8.3 Constant Local Skew with Halting Clocks

From Theorem 8.4, we know that we cannot concurrently have

- $\frac{dL_v}{dt}(t) \geq \frac{dH_v}{dt}(t)$ for all v and t,
- $\frac{dL_v}{dt}(t) \leq (1 + \mu)\frac{dH_v}{dt}(t)$ for all v and t, and
- $\mathcal{L} \leq f(d, u, \mu, \vartheta)$ for some function f, i.e., a local skew that does not depend on the network size.

In the next sections, we will address these points one by one.

In this section, we start with the first point. It entails that all logical clocks increase at rate at least 1 at all times. We now show how the other two requirements can be satisfied, by relaxing the first one to *amortized* 1-progress (see Definition 7.10, i.e., we demand that for each execution there is some $C \in \mathbb{R}_{\geq 0}$ such that for all $t' \geq t$ and all $v \in V$ it holds that

$$L_v(t') - L_v(t) \geq t' - t - C.$$

In order to prove this claim, we analyze Algorithm 6, which can be viewed as a TMP version of Algorithm 2 that maintains logical clocks. First, we prove amortized 1-progress.

Algorithm 6 This clock synchronization algorithm can be viewed as a TMP variant of Algorithm 2; replacing the "tick" messages by the messages of some synchronous algorithm (labeled by round number), this algorithm could be simulated.

```
if t = 0 (i.e., v just woke up) then
    ℓ ← 0                                          ▷ initialize L_v(0) to 0
    h ← getH()
    r ← 1                                          ▷ logical clock is initially running
    for each p ∈ {1, . . . , deg(v) do
        COUNT_p ← 0
        send "tick" to p
    end for
end if
if r = 1 and getH() − h = d then
    ℓ ← ℓ + d                                      ▷ memorize progress
    h ← getH()
    for p ∈ {1, . . . , deg(v)} do
        send "tick" on p                           ▷ inform neighbors on progress
        COUNT_p ← COUNT_p − 1                      ▷ consume neighbors' ticks
        if COUNT_p = −1 then                       ▷ made d progress since last tick on p
            r ← 0                                  ▷ stop clock until tick on p arrives
        end if
    end for
end if
if received "tick" on port p then
    COUNT_p ← COUNT_p + 1
    if r = 0 and COUNT_q ≥ 0 for all q ∈ {1, . . . , deg(v) then
        h ← getH()
        r ← 1                                      ▷ restart clock
    end if
end if
procedure getL()                                   ▷ returns L_v(t)
    if r = 1 then
        return ℓ + getH() − h                      ▷ logical clock increases at rate dH_v/dt
    else
        return ℓ                                   ▷ logical clock is halted
    end if
end procedure
```

Lemma 8.5. *In graphs of diameter D, Algorithm 6 satisfies amortized* 1-*progress with* $C = 0$.

E8.1 Prove the lemma. Hint: Review the proof of Theorem 6.18 and recall that we assume that all nodes wake up at time 0.

Lemma 8.6. *Algorithm 6 satisfies for each* $v \in V$ *that* L_v *is continuous and that* $\frac{dL_v}{dt}(t) \leq \frac{dH_v}{dt}(t)$ *at all times* t *when* r_v *does not change.*

E8.2 Prove the lemma.

Lemma 8.7. *Algorithm 6 satisfies* $\mathcal{L} \leq \max\{d, \vartheta u\}$.

Proof. Fix neighbors $v, w \in V$ and a time t. W.l.o.g., assume that $L_v(t) \geq L_w(t)$ (otherwise, flip v and w). Let i be the number of tick messages v has sent by time t, and j the number of tick messages from w it has received. By a simple induction, we have that $j \geq i - 1$ (for every "tick" after its first, v waits for a "tick" from w), $(i-1)d \leq L_v(t) \leq id$, and $L_w(t) \geq (j-1)d$ (w must reach this clock value to send j "tick" messages). If $j \geq i$ or $i = 1$, we have that $L_v(t) - L_w(t) \leq d$.

Thus, it remains to consider the case that $j = i - 1$ for $i \geq 2$. Denonte by t_r the time when v received the j-th "tick" message from w and by t_s the time it was sent. Observe that $L_w(t_s) = (j-1)d$ and $L_v(t_r) \leq (i - d)$. Because w sets r to 1 when sending the message and does not set r back to 0 before d time has passed on its hardware clock, we have that

$$
\begin{aligned}
L_w(t) &\geq L_w(t_s) + \min\{d, H_w(t) - H_w(t_s)\} && \text{Line 30} \\
&= (j-1)d + \min\{d, H_w(t) - H_w(t_s)\} \\
&= (i-2)d + \min\{d, H_w(t) - H_w(t_s)\} && j = i - 1 \\
&\geq (i-2)d + \min\{d, t - t_s\} \qquad (8.5) && \frac{dH_w}{dt} \geq 1
\end{aligned}
$$

If $t - t_s \geq d$, we again get that $L_w(t) \geq (i-1)d$ and hence $L_v(t) - L_w(t) \leq d$, so assume that $t - t_s < d$. Then

$$\begin{aligned}
&\text{adding 0} & L_v(t) - L_w(t) &= L_v(t_r) + L_v(t) - L_v(t_r) - L_w(t) \\
& & &\leq (i-1)d + L_v(t) - L_v(t_r) - L_w(t) \\
&\text{Lemma 8.6} & &\leq (i-1)d + H_v(t) - H_v(t_r) - L_w(t) \\
&\tfrac{dH_v}{dt} \leq \vartheta & &\leq (i-1)d + \vartheta(t - t_r) - L_w(t) \\
&t - t_s < d, (8.5) & &\leq d + (\vartheta - 1)(t - t_r) - (t_r - t_s) \\
&t_r - t_s \geq d - u & &\leq d + (\vartheta - 1)(t - t_s) - \vartheta(d - u) \\
&t - t_s < d & &< \vartheta u. \qquad \square
\end{aligned}$$

Theorem 8.8. *There is a clock synchronization algorithm achieving a local skew of* $\max\{d, \vartheta u\}$*, amortized* 1*-progress with* $C = 0$*, and* $\frac{dL_v}{dt}(t) \leq \frac{dH_v}{dt}(t)$ *for all times* t *and nodes* $v \in V$*.*

Proof. Follows from Lemmas 8.5 to 8.7. □

E8.3 Show that a node may stop its logical clock (i.e., continuously have $r_v = 0$) for uD time. Hint: Maximize the global skew while all message delays are $u - d$. Then have a chain of messages starting at the node that is most behind all have delay d.

E8.4 Show that this is the worst case, i.e., no node halts its logical clock for more than uD time. Hint: Consider the time when some node halts its clock after generating tick i and argue that nodes in distance r have generated their tick $i - r$ at the latest $r(d - u)$ time earlier. This requires to use that all nodes wake up at time 0 (otherwise it holds only for sufficiently large times).

8.4 Lower Bound with Arbitrary Clock Rates

We will now show that clock rates $\frac{dL_v}{dt}(t) \in \omega(\log_{1/(\vartheta-1)} D)$ do not help. That is, if $(\vartheta - 1)d < u/4$, we have that $\mathcal{L} \in \Omega(u \log_{(\log_{1/(\vartheta-1)} D)/(\vartheta-1)} D)$.

To this end, we need a technical lemma stating that, provided that we leave some slack in terms of clock drifts and message delays, we can introduce $\Omega(u)$ hardware clock skew between any pair of neighbors in an indistinguishable manner. As this follows from repetition of previous arguments, we skip the proof.

Lemma 8.9. *Let* $\mathcal{E}$ *be any execution in which hardware clock rates are at most* $1 + (\vartheta - 1)/2$ *and message delays are in the range* $(d - 3u/4, d - u/4)$*. Then, for any* $\{v, w\} \in E$ *and sufficiently large times* t*, there is an indistinguishable execution* $\mathcal{E}_v$ *such that* $L_v^{(\mathcal{E}_v)}(t) = L_v^{(\mathcal{E})}(t + u/4)$ *and* $L_w^{(\mathcal{E}_v)}(t) = L_w^{(\mathcal{E})}(t)$*.*

Proof Sketch. The general idea is to use the remaining slack of $u/2$ to hide the additional skew, and the slack in the clock rates to introduce it. We can do this as slowly as needed, just as in the proof of Lemma 7.11. Again, we can choose the clock rates according to the function $d(x)$ defined in Lemma 7.11; as v and w are neighbors here, it can only take on values of -1, 0, or 1. □

This is all we need to generalize our lower bound to arbitrarily large logical clock rates.

Theorem 8.10. *Any clock synchronization algorithm satisfying that $\frac{dH_v}{dt}(t) \leq \frac{dL_v}{dt}(t)$ for all nodes v and times t has*

$$\mathcal{L} = \Omega\left(\left(\frac{u}{4} - (\vartheta - 1)d\right) \log_{\lceil \sigma \rceil} D\right)$$

for $\sigma = \log_{1/(\vartheta-1)} D)/(\vartheta - 1)$.

Proof. Set $u' := u/2$, $d' := d - u/4$, and $\vartheta' := 1 + (\vartheta - 1)/2$. We perform the exact same construction as in Theorem 8.4, with three modifications. First, u, d, and ϑ are replaced by u', d', and ϑ'. Second, before starting the construction, we wait for sufficiently long so that Lemma 8.9 is applicable to all times when we actually "work," i.e., we let the algorithm run for the required time with hardware clock rates of 1 and message delays of $d' - u'/2$. Third, we assume that $\mu = \log_{1/(\vartheta-1)} D$ in the construction, resulting in the base of the logarithm being $\sigma' = 2\mu/(\vartheta - 1) = \Theta(\sigma)$; if we ever attempt to use this (assumed) bound on the clock rates in an inequality and it does not hold, the construction fails.

Now two things can happen. The first is that the construction succeeds. Note that we may assume that $u'/4 > (\vartheta' - 1)d'$, as otherwise $u/4 < (\vartheta - 1)d$, i.e., nothing is to show. Hence, the construction shows a lower bound of

$$\begin{aligned}\left(\frac{u'}{4} - (\vartheta' - 1)d'\right) \log_{\lceil \sigma' \rceil} D &= \left(\frac{u}{8} - \frac{(\vartheta - 1)d}{2}\right) \log_{\lceil 2\sigma \rceil} D \\ &= \Omega\left(\left(\frac{u}{4} - (\vartheta - 1)d\right) \log_{\lceil \sigma \rceil} D\right),\end{aligned}$$

i.e., the claim follows in this case.

On the other hand, if the construction fails, there is an index $i < i_{\max}$ for which (8.3) does not hold—this is the only place where we make use of the fact that logical clocks do not run faster than rate μ. Thus,

$$L_w^{(\mathcal{E}^{(i)})}(t_{i+1}) - L_w^{(\mathcal{E}^{(i)})}(t_i) > \mu(t_{i+1} - t_i)$$

for some $i < i_{\max}$. Recall that in the construction, $\text{dist}(v, w) = b^{i_{\max}-i} \geq b$ and

$$t_{i+1} - t_i = d + \left(\frac{u}{2(\vartheta - 1)} - d\right) \frac{\text{dist}(v, w)}{b} > \frac{u}{2(\vartheta - 1)} - d > \frac{u}{4(\vartheta - 1)} \geq \frac{u}{4}.$$

Hence, there must be a time $t \geq t_i$ so that

$$L_w^{(\mathcal{E}^{(i)})}\left(t + \frac{u}{4}\right) - L_w^{(\mathcal{E}^{(i)})}(t) > \frac{\mu u}{4}.$$

Let x be an arbitrary neighbor of w. By Lemma 8.9, we can construct an execution $\mathcal{E}_w$ so that

$$L_w^{(\mathcal{E}_w)}(t) = L_w^{(\mathcal{E}^{(i)})}\left(t + \frac{u}{4}\right) > L_w^{(\mathcal{E}^{(i)})}(t) + \frac{\mu u}{4}$$

and $L_x^{(\mathcal{E}_w)}(t) = L_x^{(\mathcal{E}^{(i)})}(t)$. Thus, in at least one of the executions, the local skew exceeds

$$\frac{\mu u}{8} = \frac{u}{8}\log_{1/(\vartheta-1)} D > \frac{u}{8}\log_{\mu/(\vartheta-1)} D = \Omega\left(\left(\frac{u}{4} - (\vartheta-1)d\right)\log_{\lceil\sigma\rceil} D\right). \quad \square$$

8.5 Upper Bound on the Local Skew

We now turn to proving the upper bound given in Theorem 8.28. Before presenting the algorithm, we provide some intuition by discussing why naive approaches fail.

8.5.1 Averaging Protocols

In this section, we consider a natural strategy for achieving gradient clock synchronization: trying to bring the own logical clock to the average value between the neighbors whose clocks are furthest ahead and behind, respectively. Specifically, each node can be in either *fast mode* or *slow mode*. If a node v detects that its clock is behind the average of its neighbors, it will run in fast mode, and increase its logical clock at a rate faster than its hardware clock by a factor of $1 + \mu$, for some $\mu > \vartheta - 1$. On the other hand, if v's clock is at least the average of the maximum and minimum clock value of its neighbors, it will run in slow mode, increasing its logical clock only as quickly as its hardware clock. Note that this strategy results in logical clocks that behave like "real" clocks of maximum rate $\vartheta' = 1 + \mu + (\vartheta - 1)(1 + \mu)$. If $\max\{\mu, \vartheta - 1\} \leq 1$, we have that $\vartheta' < 1 + 3\mu$. Note that for $\mu \in O(\vartheta - 1)$, these clocks are roughly as good as the original hardware clocks.

The idea of switching between fast and slow modes gives a well-defined protocol if neighboring clock values are known precisely.[12] However, ambiguity arises in the presence of uncertainty.

[12] There is one issue of pathological behavior in which nodes could switch infinitely quickly between fast and slow modes. This can be avoided by introducing a small threshold ε so that a node only changes, say, from slow to fast mode if it detects that its clock is ε time units behind the average.

To simplify our presentation of the considered synchronization algorithms, we abstract away from the individual messages and message delays for the moment. Instead, we assume that throughout an execution, each node v maintains an estimate of its neighbors' logical clocks. Specifically, for each neighbor w, v maintains a variable $\tilde{L}_w^v(t)$. The parameter δ represents the *error* in the estimates: for all $\{v, w\} \in E$ and $t \in \mathbb{R}_{\geq 0}$, we have

$$L_w(t) - \delta < \tilde{L}_w^v(t) \leq L_w(t)$$

When the node v is clear from context, we will omit the superscript v, and simply write $\tilde{L}_w$. Assuming that $\mu, \vartheta - 1 = O(1)$, in Section 8.5.4 we discuss how to obtain such estimates satisfying that $\delta = O(u + \mu d)$.

We consider two natural ways of dealing with the uncertainty. Set $L_v^{\max}(t) := \max_{\{v,w\} \in E}\{L_w(t)\}$ and $L_v^{\min}(t) := \min_{\{v,w\} \in E}\{L_w(t)\}$.

Aggresive strategy: each v computes an *upper bound* on the average between $L_v^{\max}$ and $L_v^{\min}$, and determines whether to run in fast or slow mode based on this upper bound;

Conservative strategy: each v computes a *lower bound* on the average between $L_v^{\max}$ and $L_v^{\min}$ and determines the mode accordingly.

We will see that both strategies give bad results, but for opposite reasons.

Aggressive Averaging Here we analyze the aggressive averaging protocol described above. Specifically, each node $v \in V$ computes an upper bound on the average of its neighbors' logical clock values:

$$\tilde{L}_v^{\text{up}}(t) := \frac{\max_{\{v,w\} \in E}\{\tilde{L}_w\} + \min_{w \in N_v}\{\tilde{L}_w\}}{2} + \delta \geq \frac{L_v^{\max} + L_v^{\min}}{2}\,.$$

The algorithm then increases the logical clock of v at a rate of $\frac{dH_v}{dt}(t)$ if $L_v(t) > \tilde{L}_v^{\text{up}}(t)$, and a rate of $(1 + \mu)\frac{dH_v}{dt}(t)$ otherwise. We show that the algorithm performs poorly for any choice of $\mu \geq 0$.

Lemma 8.11. *Consider the aggressive averaging protocol on a path network of diameter D, i.e., $V = [D + 1]$ and $E = \{\{i, i + 1\} \mid i \in [D]\}$. Then $\mathcal{L} \geq (2D - 1)\delta$, even if $H_v(0) = L_v(0) = 0$ for all $v \in V$.*

Proof. Throughout the execution, we will assume that all clock estimates are correct: for all $v \in V$ and $\{v, w\} \in E$, we have $\tilde{L}_w(t) = L_w(t)$. This means that for all $i \in \{1, 2, \ldots, D - 1\}$, $\tilde{L}_{v_i}^{\text{up}}(t) = (L_{i-1}(t) + L_{i+1}(t))/2 + \delta$, whereas $\tilde{L}_0^{\text{up}}(t) = L_1(t) + \delta$ and $\tilde{L}_D^{\text{up}} = L_{D-1}(t) + \delta$. Thus, all nodes immediately go into fast mode in order to catch up in case they underestimate their neighbors' clock values.

Initially, the hardware clock rate of node i is $1 + \frac{i(\vartheta-1)}{D}$. Let us see what this means for the logical clocks. While nodes are running fast, skew keeps building up, but the property that $L_{v_i}(t) = (L_{v_{i+1}}(t) - L_{v_{i-1}}(t))$ is maintained at nodes $i \in \{1, \dots, D\}$. In this state, node 0—despite running fast—has no way of catching up to node 1. However, at time $\tau_0 := \frac{\delta D}{(1+\mu)(\vartheta-1)}$ we would have that $L_D(\tau_0) = L_{D-1}(\tau_0) + \delta = \tilde{L}_D^{\mathrm{up}}(\tau_0)$ and node D would stop running fast. We set $t_0 := \tau_0 - \varepsilon$ for some arbitrarily small $\varepsilon > 0$ and set $\frac{dH_D}{dt}(t) := \frac{dH_{D-1}}{dt}(t)$ for all $t \geq t_0$. Thus, all nodes would remain in fast mode until the time $\tau_1 := t_0 + \frac{2\delta D}{(1+\mu)(\vartheta-1)}$ when we had $L_{D-1}(\tau_1) = \tilde{L}_{D-1}^{\mathrm{up}}(\tau_1)$. We set $t_1 := \tau_1 - \varepsilon$ and proceed with this construction inductively. Note that, with every hop, the local skew increases by (almost) 2δ, as this is the additional skew that L_i must build up to L_{i-1} when $L_{i+1} - L_i = L_i - L_{i-1}$ in order to increase $\tilde{L}_i^{\mathrm{up}} - L_i$ by δ, i.e., for node i to stop running fast. As ε is arbitrarily small, we build up a local skew that is arbitrarily close to $(2D-1)\delta$. □

This lemma can be generalized to arbitrary graphs, by taking two nodes $v, w \in V$ in distance D and using the function $d(x) = \mathrm{dist}(x, v) - \mathrm{dist}(x, w)$, just as in Lemma 7.11.

Note that the algorithm is also bad in that the above execution results in a global skew of $\Omega(\delta D^2)$. Slight modifications of the algorithm can guarantee better global skew, but similar algorithms will still have large local skew.

Conservative Averaging Let us be more careful now. Each node $v \in V$ computes a *lower* bound on the average of its neighbors' logical clock values:

$$\tilde{L}_v^{\mathrm{up}}(t) = \frac{\max_{w \in N_v}\{\tilde{L}_w\} + \min_{w \in N_v}\{\tilde{L}_w\}}{2} \leq \frac{L_{N_v}^{\max} + L_{N_v}^{\min}}{2}.$$

The algorithm then increases the logical clock of v at a rate of $\frac{dH_v}{dt}(t)$ if $L_v(t) > \tilde{L}_v^{\mathrm{up}}(t)$, and a rate of $(1+\mu)\frac{dH_v}{dt}(t)$ otherwise. Again, the algorithm fails to achieve a small local skew.

Lemma 8.12. *Consider the conservative averaging protocol on a path network of diameter D, i.e., $V = [D+1]$ and $E = \{\{i, i+1\} \mid i \in [D]\}$. Then $\mathcal{L} \geq (2D-1)\delta$, even if $H_v(0) = L_v(0) = 0$ for all $v \in V$.*

Proof Sketch. We use the same initial hardware clock rates as for the aggressive strategy, but now for each $v \in V$, $\{v, w\} \in E$, and time t, we rule that $\tilde{L}_w(t) = L_w(t) - \delta + \varepsilon$ for some arbitrarily small $\varepsilon > 0$. Thus, all nodes are initially in slow mode. We inductively change hardware clock speeds just before nodes would switch to fast mode, building up the exact same skews (up to terms in ε) between logical clocks as in the previous execution. The only difference is that now it does not depend on μ how long this takes! □

8.5.2 GCS Algorithm

We have seen that both the aggressive and the conservative strategy do not result in a proper response to the gobal distribution of clock values. The main issue is that these approaches ignore the measurement error encapsulated in δ. The underlying idea of distributing skews evenly over paths is sound, but requires to discretize "skew levels."

The high-level strategy of the (functional) GCS algorithm is as follows. As with the naive algorithms from Section 8.5.1, at each time a node can be either in *slow mode* or *fast mode*. In slow mode, a node v will increase its logical clock at rate $\frac{dH_v}{dt}(t)$. In fast mode, v will increase its logical clock at rate $(1+\mu)\frac{dH_v}{dt}(t)$. The parameter μ must be chosen large enough for nodes whose logical clocks are behind to be able to catch up to other nodes, i.e., $\mu > \vartheta - 1$. The conditions for a node to switch from slow to fast or vice versa are simple, though perhaps unintuititve. In what follows, we first describe "ideal" conditions to switch between modes. In the ideal behavior, each node knows exactly the logical clock values of its neighbors. Since the actual algorithm only has access to estimates of neighboring clocks, we then describe fast and slow triggers for switching between modes that can be implemented in our model for GCS. We conclude the section by proving that the triggers do indeed implement the conditions.

Fast and Slow Conditions. Here we define conditions under which a node should be in fast mode and slow mode. The two conditions are mutually exclusive (i.e., a node cannot simulatenously satisfy both), but it could be that a node satisfies neither condition. The conditions are defined in terms of a parameter κ, whose value will be determined later.

Definition 8.13 (**FC**: Fast Mode Condition)**.** *We say that a node $v \in V$ satisfies the* fast mode condition (**FC**) *at time $t \in \mathbb{R}_{\geq 0}$ if there exists $s \in \mathbb{N}_{>0}$ such that:*

FC-1 $\exists\{v,x\} \in E\colon L_x(t) - L_v(t) \geq 2s\kappa$,

FC-2 $\forall\{v,y\} \in E\colon L_v(t) - L_y(t) \leq 2s\kappa$.

Informally, **FC-1** says that v has a neighbor x whose logical clock is significantly ahead of $L_v(t)$, while **FC-2** stipulates that none of v's neighbors' clocks is too far behind $L_v(t)$. In particular, if **FC** is satisfied with neighbor x fulfilling **FC-1**, then the local skew across $\{v,x\}$ is at least $2s\kappa$, where L_x is at least $2s\kappa$ time units ahead of L_v. On the other hand, **FC-2** implies that none of v's neighbors are more than $2s\kappa$ behind v. Therefore, v can decrease the maximum skew across its incident edges by increasing its logical clock.

The slow mode condition below is dual to **FC**. It gives sufficient conditions under which v could decrease the maximum skew across its incedent edges by decreasing its logical clock.

Definition 8.14 (**SC**: Slow Mode Condition)**.** *We say that a node $v \in V$ satisfies the* slow mode condition **SC**) *at time $t \in \mathbb{R}_{\geq 0}$ if there exists $s \in \mathbb{N}_{>0}$ such that:*

SC-1 $\exists \{v, x\} \in E\colon L_v(t) - L_x(t) \geq (2s - 1)\kappa,$

SC-2 $\forall \{v, y\} \in E\colon L_y(t) - L_v(t) \leq (2s - 1)\kappa.$

There is a slight asymmetry in the definitions of **FC** and **SC** in the coefficient of κ appearing on the right hand side of the expressions above. The **FC** conditions bound the differences in logical clocks by $2s\kappa$—an even multiple of s—while the **SC** conditions give odd multiples of s. This discrepancy between the definitions of **FC** and **SC** ensures that a node cannot simultaneously satisfy both conditions.

We say that an algorithm *implements the F/S conditions* if for every execution, every node v, and every time t we have:

- if v satisfies **FC** at time t, then v is in fast mode at time t,
- if v satisfies **SC** at time t, then v is in slow mode at time t.

At this point we have not shown that *any* algorithm can implement the F/S conditions. Indeed, in our model v does not know its neighbors' logical clock values precisely at any time, so it cannot directly check if **FC** or **SC** is satisfied. However, we will show that for an appropriate choice of κ, there is a simple algorithm that implements the F/S conditions. Interestingly, the analysis we give applies to *any* algorithm implementing the F/S conditions, and not just for the particular implementation we describe.

Fast and Slow Triggers. While the fast and slow mode conditions described in the previous section are well-defined (and mutually exclusive), uncertainty on neighbors' clock values prevents an algorithm from checking the conditions directly. Here we define corresponding *triggers* that our computational model does allow us to check. As before, we assume that for each node v and neighbor $w \in N_v$, v maintains a clock estimate $\tilde{L}_w^v$ satisfying

$$L_w(t) \geq \tilde{L}_w^v(t) < L_w(t) - \delta. \tag{8.6}$$

For convenience, we omit the superscript when v is clear from context.

Fix a node v, and suppose that v satisfies **FC** at time t. Let x be a node for which v satisfies **FC-1**, i.e., $L_x(t) - L_v(t) \geq 2s\kappa$. Since v's estimate of $L_x(t)$ is generally smaller than $L_x(t)$, it could be the case that $\tilde{L}_x(t) - L_v(t) < 2s\kappa$, so that v does see that **FC-1** is satisfied. Since $\tilde{L}_w(t) \geq L_w(t) - \delta$, v *might* satisfy

FC-1 if $\tilde{L}_w(t) - L_v(t) \geq 2s\kappa - \delta$. Thus, in order to ensure that v switches to fast mode whenever **FC** is satisfied, we should relax the conditions **FC** to ensure that v switches to fast mode whenever its estimates indicate that **FC** could be satisfied. Thus we define the following *triggers*.

Definition 8.15 (**FT**: Fast Mode Trigger)**.** *We say that $v \in V$ satisfies the* fast mode trigger (**FT**) *at time $t \in \mathbb{R}_{\geq 0}$ if there exists an integer $s \in \mathbb{N}_{>0}$ such that:*

FT-1 $\exists \{v, x\} \in E : \tilde{L}_x(t) - L_v(t) > 2s\kappa - \delta,$

FT-2 $\forall \{v, y\} \in E : L_v(t) - \tilde{L}_y(t) < 2s\kappa + \delta.$

Definition 8.16 (**ST**: Slow Mode Trigger)**.** *We say that a node $v \in V$ satisfies the* slow mode trigger (**ST**) *at time $t \in \mathbb{R}_{\geq 0}$ if there exists $s \in \mathbb{N}_{>0}$ such that:*

ST-1 $\exists \{v, x\} \in E : L_v(t) - \tilde{L}_x(t) \geq (2s - 1)\kappa,$

ST-2 $\forall \{v, y\} \in E : \tilde{L}_y(t) - L_v(t) \leq (2s - 1)\kappa.$

Note that we do *not* add the extra δ slack in the definition of **ST**, as we did in **FT**. This is because we assume that the uncertainty in neighboring clock estimates is one-sided: for all $\{v, w\} \in E$ and times t we have $\tilde{L}^v_w(t) \leq L_w(t)$. Thus, if a node satisfies **SC,** its neighboring clock estimates automatically satisfy **ST**.

Before we formally describe the GCS algorithm, we give two preliminary results about the fast and slow mode triggers. The first result asserts that for a suitable choice of κ, **FT** and **ST** cannot simultaneously be satisfied by the same node. The second shows that for the same choice of κ, **FT** and **ST** implement **FC** and **SC**, respectively. That is, if the fast (resp. slow) mode condition is satisfied, then the fast (resp. slow) mode trigger is also satisfied.

Lemma 8.17. *Suppose $\kappa \geq \delta$. Then no node $v \in V$ can simultaneously satisfy* **FT** *and* **ST**.

Proof. Suppose v satisfies **FT**. That is, there is some $s \in \mathbb{N}_{>0}$ and $\{v, x\} \in E$ such that $\tilde{L}_x(t) - L_v(t) \geq 2s\kappa - \delta$, and for all $\{v, y\} \in E$ we have $L_v(t) - \tilde{L}_y(t) < 2s\kappa + \delta$. Consider $s' \in \mathbb{N}_{>0}$. If $s' > s$, then for all $\{v, x\} \in E$ we have

$$L_v(t) - \tilde{L}_x(t) < 2s\kappa - \delta \leq (2s' - 1)\kappa, \tag{8.6}$$

so **ST-1** is not satisfied for s'. If $s' \leq s$, then there is some $\{v, y\} \in E$ satisfying

$$\tilde{L}_y(t) - L_v(t) > 2s\kappa - \delta \geq (2s' - 1)\kappa,$$

so **ST-2** is not satisfied for s'. Hence, **ST** is not satisfied. □

For the remainder of our analysis, we assume that the condition that $\kappa > \delta$ is met

Lemma 8.18. *Suppose $v \in V$ satisfies* **FC** *(resp.* **SC)** *at time t. Then v satisfies* **FT** *(resp.* **SC***) at time t.*

Proof. Suppose **FC** holds (at time t). Then, by (8.6), there is some $s \in \mathbb{N}_{>0}$ such that

$$\exists\{v,x\} \in E\colon \tilde{L}_x(t) - L_v(t) \geq L_x(t) - \delta - L_v(t) \geq 2s\kappa - \delta,$$

and

$$\forall\{v,y\} \in E\colon L_v(t) - \tilde{L}_y(t) \leq L_v(t) - L_y(t) + \delta \leq 2s\kappa + \delta.$$

Thus **FT** holds. Similarly, if **SC** holds, (8.6) yields

$$\exists\{v,x\} \in E\colon L_v(t) - \tilde{L}_x(t) \geq L_v(t) - L_x(t) \geq (2s-1)\kappa$$

and

$$\forall\{v,y\} \in E\colon \tilde{L}_y(t) - L_x(t) \leq L_y(t) - L_v(t) \leq (2s-1)\kappa,$$

for some $s \in \mathbb{N}_{>0}$, establishing **ST**. □

The Algorithm. We now describe the GCS algorithm, whose pseudocode is given in Algorithm 7. To focus on the key ideas of the analysis, we make another simplifying assumption: Instead of analyzing the global skew, for now we assume that it is bounded by some parameter $\mathcal{G}$. We will prove a bound on $\mathcal{G}$ later, in Section 8.5.3. Each node v initializes its logical clock to its hardware clock value. It keeps checking the slow mode trigger is satisfied. During such times, it increases its logical clock at the hardware clock rate. By Lemma 8.17, only when the slow mode trigger is not satisfied, the fast mode trigger might hold. By Lemma 8.18, switching to fast mode whenever the slow mode trigger does not hold is hence sufficient to implement the fast and slow conditions. In fast mode, v increases its logical clock at a rate of $(1+\mu)\frac{dH_v}{dt}(t)$. Despite the algorithm's simplicity, its analysis (presented in the following section) is rather delicate.

8.5.3 Analysis of the GCS Algorithm

We now show that the GCS algorithm (Algorithm 7) indeed achieves a small local skew. To this end, we analyze the *average* skew over paths in G of various lengths. For long paths of $\Omega(D)$ hops, we will simply exploit that $\mathcal{G}$ bounds the skew between *any* pair of nodes. For successively shorter paths, we inductively show that the average skew between endpoints cannot increase too quickly: reducing the length of a path by factor σ can only increase the skew between endpoints by an additive constant term. Thus, paths of constant length

Algorithm 7 GCS algorithm

```
if v just woke up, i.e., t = 0 then
    ℓ ← getH()
    h ← getH()
    if ST then
        r ← 1                                   ▷ v is in slow mode
    else
        r ← 1 + μ                               ▷ v is in fast mode
    end if
end if
if ST stops to hold then
    ℓ ← getL()                                  ▷ always keep track of clock progress
    h ← getH()
    r ← 1 + μ                                   ▷ v is in fast mode
end if
if ST starts to hold then
    ℓ ← getL()                                  ▷ always keep track of clock progress
    h ← getH()
    r ← 1                                       ▷ v is in slow mode
end if
procedure getL()                                ▷ returns L_v(t)
    return ℓ + r(getH() − h)                    ▷ logical clock increases at rate r dH_v/dt
end procedure
```

(in particular edges) can only have a(n average) skew that is logarithmic in the network diameter.

To simplify the analysis, we will assume that the logical clocks are differentiable. Note that this is not the case at times when r changes. However, they can be approximated arbitrarily well by differentiable functions whose derivative is bounded from below by $\frac{dH_v}{dt}$ and from above by $(1+\mu)\frac{dH_v}{dt}$. Even with this simplification, our analysis requires a technical lemma formalizing an intuitive statement: If the maximum of a finite set of functions satisfies that whenever one of the functions attains the maximum, its derivative is bounded by r, then the growth of the maximum is also bounded by r. For completeness, we prove this lemma at the end of the chapter.

Leading Nodes We start by showing that skew cannot build up too quickly. This is captured by the following functions.

Definition 8.19 (Ψ and Leading Nodes)**.** *For each $v \in V$, $s \in \mathbb{N}_{>0}$, and $t \in \mathbb{R}_{\geq 0}$, we define*

$$\Psi_v^s(t) = \max_{w \in V}\{L_w(t) - L_v(t) - (2s-1)\kappa\,\mathrm{dist}(v,w)\},$$

where $\mathrm{dist}(v,w)$ *denotes the distance between v and w in G. Moreover, set*

$$\Psi^s(t) = \max_{w \in V}\{\Psi_w^s(t)\}.$$

Finally, we say that $w \in V$ is a leading node *if there is some $v \in V$ satisfying*

$$\Psi_v^s(t) = L_w(t) - L_v(t) - (2s-1)\kappa\,\mathrm{dist}(v,w) > 0.$$

Observe that any bound on Ψ^s implies a corresponding bound on $\mathcal{L}$: if $\Psi^s(t) \leq \alpha$, then in particular, for any adjacent nodes v, w we have $L_w(t) - L_v(t) - (2s-1)\kappa \leq \Psi^s(t) \leq \alpha$. Therefore, $\Psi^s(t) \leq \alpha \implies \mathcal{L} \leq (2s-1)\kappa + \alpha$. Our analysis will show that in general, $\Psi^s(t) \leq \mathcal{G}/\sigma^s$ for every $s \in \mathbb{N}_{>0}$ and all times t. In particular, Theorem 8.28 will follow by considering $s = \lceil \log_\sigma \mathcal{G}/\delta \rceil$.

Note that the definition of Ψ_v^s is closely related to the definition of **SC**. In fact, the following lemma shows that if w is a leading node, then w satisfies **SC**. As a result Ψ^s cannot increase quickly, because leading nodes are always in slow mode for any algorithm implementing the F/S conditions. This behavior allows nodes in fast mode to catch up to leading nodes.

Lemma 8.20 (Leading Lemma)**.** *Suppose $w \in V$ is a leading node at time t. Then w satisfies* **SC** *and* **ST**.

Proof. By Lemma 8.18, if w satisfies **SC,** then w also satisfies **ST**. Thus, it suffices to prove that w satisfies **SC**. As w is a leading node at time t, there are $s \in \mathbb{N}_{>0}$ and $v \in V$ satisfying

$$\Psi_v^s(t) = L_w(t) - L_v(t) - (2s-1)\kappa\,\mathrm{dist}(v,w) > 0.$$

In particular, $L_w(t) > L_v(t)$, so $w \neq v$. For any $y \in V$, we have

$$\begin{aligned} L_w(t) - L_v(t) - (2s-1)\kappa\,\mathrm{dist}(v,w) &= \Psi_v^s(t) \\ &\geq L_y(t) - L_v(t) - (2s-1)\kappa\,\mathrm{dist}(y,w). \end{aligned}$$

Rearranging this yields

$$L_w(t) - L_y(t) \geq (2s-1)\kappa(\mathrm{dist}(v,w) - \mathrm{dist}(y,w)).$$

In particular, for any $\{v, y\} \in E$, $\mathrm{dist}(v,w) \geq \mathrm{dist}(y,w) - 1$ and hence

$$L_y(t) - L_w(t) \leq (2s-1)\kappa,$$

i.e., **SC-2** holds at w.

Now consider $\{v,x\} \in E$ so that $\operatorname{dist}(x,w) = \operatorname{dist}(v,w) - 1$. Such a node exists because $v \neq w$. We obtain

$$L_w(t) - L_y(t) \geq (2s-1)\kappa,$$

showing **SC-1**. By Lemma 8.18, w then also satisfies **ST** at time t. □

This can readily be translated into a bound on the growth of Ψ_w^s whenever it is positive. It states that Ψ_w^s can never grow faster than at rate $\vartheta - \frac{dL_w}{dt}$. Note that this means that Ψ_w^s is actually guaranteed to decrease whenever it is positive and w is in fast mode, because then $\frac{dL_w}{dt} = (1+\mu)\frac{dH_w}{dt} \geq 1 + \mu > \vartheta$.

Lemma 8.21 (Wait-up Lemma)**.** *Suppose $w \in V$ satisfies $\Psi_w^s(t) > 0$ for all $t \in (t_0, t_1]$. Then*

$$\Psi_w^s(t_1) \leq \Psi_w^s(t_0) - (L_w(t_1) - L_w(t_0)) + \vartheta(t_1 - t_0).$$

Proof. Fix $w \in V$, $s \in \mathbb{N}_{>0}$ and $(t_0, t_1]$ as in the hypothesis of the lemma. For $v \in V$ and $t \in (t_0, t_1]$, define the function $f_v(t) = L_v(t) - (2s-1)\delta \operatorname{dist}(v,w)$. Observe that

$$\max_{v \in V}\{f_v(t)\} - L_w(t) = \Psi_w^s(t).$$

Moreover, for any v for which $f_v(t)$ attains this maximum, i.e., $f_v(t) = L_w(t) + \Psi_w^s(t)$, we have that $L_v(t) - L_w(t) - (2s-1)\kappa \operatorname{dist}(v,w) = \Psi_w^s(t) > 0$. Thus, v is a leading node and Lemma 8.20 shows that v is in slow mode at time t. As (we assume that) logical clocks are differentiable, so is f_v, and it follows that $\frac{df_v}{dt}(t) = \frac{dL_v}{dt}(t) = \frac{dH_v}{dt} \leq \vartheta$ for each $v \in V$ and time $t \in (t_0, t_1]$ satisfying $f_v(t) = \max_{x \in V}\{f_x(t)\}$. Therefore, we can apply Lemma 8.33 to show that

$$\begin{aligned}\Psi_w^s(t_1) - \Psi_w^s(t_0) &= \max_{v \in V}\{f_v(t_1)\} - \max_{v \in V}\{f_v(t_0)\} - (L_w(t_1) - L_w(t_0)) && \text{adding 0}\\ &\leq \vartheta(t_1 - t_0) - (L_w(t_1) - L_w(t_0)), && \text{Lemma 8.33}\end{aligned}$$

which gives the desired result. □

As logical clocks increase at least as fast as hardware clocks, this means that Ψ^s can never increase faster than at rate $\vartheta - 1$.

Corollary 8.22. *For all $w \in V$ and $s \in \mathbb{N}_{>0}$ it holds that*

$$\Psi_w^s(t_1) \leq \Psi_w^s(t_0) + (\vartheta - 1)(t_1 - t_0).$$

Proof.

$$\begin{aligned}
\Psi_w^s(t_1) &\le \Psi_w^s(t_0) - (L_w(t_1) - L_w(t_0)) + \vartheta(t_1 - t_0) && \text{Lemma 8.21}\\
&\le \Psi_w^s(t_0) - (H_w(t_1) - H_w(t_0)) + \vartheta(t_1 - t_0) && \tfrac{dL_w}{dt} \ge \tfrac{dH_w}{dt}\\
&\le \Psi_w^s(t_0) + (\vartheta - 1)(t_1 - t_0). && \tfrac{dH_w}{dt} \ge 1
\end{aligned}$$ □

Trailing Nodes As $L_w(t_1) - L_w(t_0) \ge t_1 - t_0$ at all times, Lemma 8.25 implies that Ψ^s cannot grow faster than at rate $\vartheta - 1$ when $\Psi^s(t) > 0$. This means that nodes whose clocks are far behind leading nodes can catch up, so long as the slow nodes are in fast mode. Our next task is to show that "trailing nodes" always run in fast mode so that they are never too far behind leading nodes. The approach to showing this is similar to the one for Lemma 8.21, where now we need to exploit the fast mode condition **FC**.

Definition 8.23 (Trailing Nodes)**.** *We say that $w \in V$ is a* trailing node *at time t, if there exists $s \in \mathbb{N}_{>0}$ and a node $v \in V$ such that*

$$L_v(t) - L_w(t) - 2s\kappa\,\mathrm{dist}(v, w) = \max_{x \in V}\{L_v(t) - L_x(t) - 2s\kappa\,\mathrm{dist}(v, x)\} > 0\,.$$

Lemma 8.24 (Trailing Lemma)**.** *Suppose $w \in V$ is a trailing node at time t. Then w satisfies* **FC** *and* **FT**.

Proof. By Lemma 8.18, if w satisfies **FC,** then w also satisfies **FT**. Thus, it suffices to prove that w satisfies **FC**. Let s and v satisfy

$$L_v(t) - L_w(t) - 2s\kappa\,\mathrm{dist}(v, w) = \max_{x \in V}\{L_v(t) - L_x(t) - 2s\kappa\,\mathrm{dist}(v, x)\} > 0.$$

In particular, $L_v(t) > L_w(t)$, implying that $v \ne w$. For $y \in V$, we have

$$L_v(t) - L_w(t) - 2s\kappa\,\mathrm{dist}(v, w) \ge L_v(t) - L_y(t) - 2s\kappa\,\mathrm{dist}(v, y).$$

Thus, for all $\{w, y\} \in E$,

$$L_y(t) - L_w(t) + 2s\kappa(\mathrm{dist}(v, y) - \mathrm{dist}(v, w)) \ge 0.$$

It follows that

$$\forall \{v, y\} \in E\colon L_w(t) - L_y(t) \le 2s\kappa,$$

i.e., **FC-2** holds. As $v \ne w$, there is some $\{v, x\} \in E$ with $\mathrm{dist}(v, x) = \mathrm{dist}(v, w) - 1$. We obtain that

$$\exists \{v, x\} \in E\colon L_y(t) - L_w(t) \ge 2s\kappa,$$

showing **FC-1**. □

Using Lemma 8.24, we can show that if $\Psi_w^s(t_0) > 0$, w will eventually catch up. How long this takes can be expressed in terms of $\Psi^{s-1}(t_0)$, or, if $s = 1$, $\mathcal{G}$.

Lemma 8.25 (Catch-up Lemma)**.** *Let $s \in \mathbb{N}_{>0}$ and t_0, t_1 be times. Suppose that*

$$t_1 \geq \begin{cases} t_0 + \frac{\mathcal{G}(t_0)}{\mu} & \text{if } s = 1 \\ t_0 + \frac{\Psi^{s-1}(t_0)}{\mu} & \text{otherwise.} \end{cases}$$

Then, for any $w \in V$,

$$L_w(t_1) - L_w(t_0) \geq t_1 - t_0 + \Psi_w^s(t_0).$$

Proof. Choose $v \in V$ such that

$$\Psi_w^s(t_0) = L_v(t_0) - L_w(t_0) - (2s-1)\kappa \operatorname{dist}(v, w) > 0.$$

Define $f_x(t) \coloneqq L_v(t_0) + (t - t_0) - L_x(t) - (2s-2)\kappa \operatorname{dist}(v, x)$ for $x \in V$ and observe that

$$\begin{aligned}
\Psi_w^s(t_0) &= L_v(t_0) - L_w(t_0) - (2s-1)\kappa \operatorname{dist}(v, w) && \text{choice of } v \\
&\leq L_v(t_0) - L_w(t_0) - (2s-2)\kappa \operatorname{dist}(v, w) && \kappa \operatorname{dist}(v, w) \geq 0 \\
&= L_v(t_0) + (t_0 - t_0) - L_w(t_0) - (2s-2)\kappa \operatorname{dist}(v, w) && \text{adding } 0 \\
&= f_w(t_0). \qquad (8.7) && \text{def. of } f_w(t_0)
\end{aligned}$$

Hence, if $\max_{x \in V}\{f_x(t)\} \leq 0$ for some $t \in [t_0, t_1]$, then

$$\begin{aligned}
L_w(t_1) - L_w(t) - (t_1 - t) \geq 0 \geq f_w(t)& \\
&= L_v(t_0) + (t - t_0) - L_w(t) - (2s-2)\kappa \operatorname{dist}(v, w) && \text{def. of } f_w(t) \\
&= f_w(t_0) + (t - t_0) - (L_w(t) - L_w(t_0)) && \text{def. of } f_w(t_0) \\
&\geq \Psi_w^s(t_0) + (t - t_0) - (L_w(t) - L_w(t_0)), && (8.7)
\end{aligned}$$

which can be rearranged into the conclusion of the lemma.

To show this, consider any time $t \in [t_0, t_1]$ when $\max_{x \in V}\{f_x(t)\} > 0$ and let $y \in V$ be any node such that $\max_{x \in V}\{f_x(t)\} = f_y(t)$. Then y is trailing, as

$$\begin{aligned}
&\max_{x \in V}\{L_v(t) - L_x(t) - (2s-2)\kappa \operatorname{dist}(v, x)\} \\
&= L_v(t) - L_v(t_0) - (t - t_0) + \max_{x \in V}\{f_x(t)\} && \text{def. of } f_x(t) \\
&= L_v(t) - L_v(t_0) - (t - t_0) + f_y(t) && \text{choice of } y \\
&= L_v(t) - L_y(t) - (2s-2)\kappa \operatorname{dist}(v, y), && \text{def. of } f_y(t)
\end{aligned}$$

which is positive because

$$\begin{aligned} L_v(t) - L_v(t_0) - (t - t_0) + \max_{x \in V}\{f_x(t)\} &> L_v(t) - L_v(t_0) - (t - t_0) && \max > 0 \\ &\geq H_v(t) - H_v(t_0) - (t - t_0) && \tfrac{dL_v}{dt} \geq \tfrac{dH_v}{dt} \\ &\geq 0. && \tfrac{dH_v}{dt} \geq 1 \end{aligned}$$

Thus, by Lemma 8.24 y satisfies **FT**, which by Lemma 8.17 entails that y does not satisfy **ST**. Accordingly, y is in fast mode, which by our assumption that logical clocks are differentiable implies that

$$\frac{df_y}{dt}(t) = 1 - \frac{dL_y}{dt}(t) = 1 - (1+\mu)\frac{dH_y}{dt}(t) \leq -\mu. \qquad \tfrac{dH_y}{dt} \geq 1$$

Now assume for contradiction that $\max_{x \in V}\{f_x(t)\} > 0$ for all $t \in [t_0, t_1]$. Then

$$\begin{aligned} \max_{x \in V}\{f_x(t_0)\} &> -\left(\max_{x \in V}\{f_x(t_1)\} - \max_{x \in V}\{f_x(t_0)\}\right) && \max > 0 \\ &\geq \mu(t_1 - t_0). && \text{Lemma 8.33} \end{aligned} \tag{8.8}$$

If $s = 1$,

$$\max_{x \in V}\{f_x(t_0)\} = \max_{x \in V}\{L_v(t_0) - L_x(t_0)\} \leq \mathcal{G}(t_0) \leq \mu(t_1 - t_0), \qquad \text{precondition of lemma}$$

contradicting (8.8). If $s > 1$,

$$\begin{aligned} \max_{x \in V}\{f_x(t_0)\} &= \max_{x \in V}\{L_v(t_0) - L_x(t_0) - (2s-2)\kappa \operatorname{dist}(v,x)\} && \text{def. of } f_x(t_0) \\ &\leq \max_{x \in V}\{L_v(t_0) - L_x(t_0) - (2s-3)\kappa \operatorname{dist}(v,x)\} && \kappa \operatorname{dist}(v,x) \geq 0 \\ &\leq \Psi^{s-1}(t_0) && \text{def. of } \Psi^{s-1} \\ &\leq \mu(t_1 - t_0). && \text{precondition of lemma} \end{aligned}$$

As this contradicts (8.8) as well, the claim of the lemma follows. □

Bound on the Local Skew As in Chapter 7, we need to assume that there is some initial degree of synchronization to show bounds on the skew that hold at all times.

Lemma 8.26. *Assume that $H_v(0) - H_w(0) \leq \kappa$ for all $\{v, w\} \in E$. Then $\Psi^s(0) = 0$ for all $s \in \mathbb{N}_{>0}$.*

Proof. We have that

$$\begin{aligned}
\Psi^s(0) &= \max_{v,w\in V}\{L_w(0) - L_v(0) - (2s-1)\kappa\,\mathrm{dist}(v,w)\} && \text{def. of } \Psi^s\\
&\le \max_{v,w\in V}\{L_w(0) - L_v(0) - \kappa\,\mathrm{dist}(v,w)\} && s\in\mathbb{N}_{>0},\ \kappa\,\mathrm{dist}(v,w)\ge 0\\
&= \max_{v,w\in V}\{H_w(0) - H_v(0) - \kappa\,\mathrm{dist}(v,w)\} && L_x(0)=H_x(0) \text{ for all } x\in V\\
&= 0,
\end{aligned}$$

where the last step uses that $H_v(0) - H_w(0) \le \kappa\,\mathrm{dist}(v,w)$ by applying the precondition that $H_x(0) - H_y(0) \le \kappa$ for all $\{x,y\} \in E$ to the edges of a shortest path from v to w. □

We now can link Lemmas 8.21 and 8.25 together to show that Ψ^s is bounded: Lemma 8.21 tells us that Ψ^s cannot grow too fast, and Lemma 8.25 shows that after a certain time we are guaranteed that the trailing nodes catch up, putting a stop to this growth. As the time that Ψ^s might grow depends on how large Ψ^{s-1} (or $\mathcal{G}$ for $s=1$) is, we get an exponential decrease of the bounds on Ψ^s as function of s, with base σ.

Lemma 8.27. *Assume that $H_v(0) - H_w(0) \le \kappa$ for all $\{v,w\}\in E$. Then, for all $s\in\mathbb{N}_{>0}$, Algorithm 7 guarantees $\Psi^s(t) \le \mathcal{G}/\sigma^s$, where $\sigma = \mu/(\vartheta-1)$.*

Proof. Suppose for contradiction that the statement of the theorem is false. Let $s\in\mathbb{N}_{>0}$ be minimal such that there is a time t_1 for which $\Psi^s(t_1) = \mathcal{G}/\sigma^s + \varepsilon$ for some sufficiently small $\varepsilon > 0$ (because $\Psi^s(0) = 0$ by Lemma 8.26 and Ψ^s is continuous, such a time exists). Thus, there is some $w \in V$ such that

$$\Psi^s_w(t_1) = \Psi^s(t_1) = \frac{\mathcal{G}}{\sigma^s} + \varepsilon. \tag{8.9}$$

Set $t_0 := \max\{t_1 - \mathcal{G}/(\mu\sigma^{s-1}), 0\}$. Consider the time $t' \in [t_0, t_1]$ that is minimal with the property that $\Psi^s_w(t) > 0$ for all $t \in (t', t_1]$ (by continuity of Ψ^s_w such a time exists). $\Psi^s_w(t')$ cannot be 0, as otherwise

$$\begin{aligned}
\frac{\mathcal{G}}{\sigma^s} + \varepsilon &= \Psi^s_w(t_1) && (8.9)\\
&\le (\vartheta-1)(t_1 - t') && \text{Corollary 8.22}\\
&\le (\vartheta-1)(t_1 - t_0) && \text{def. of } t'\\
&\le \frac{\vartheta-1}{\mu}\cdot\frac{\mathcal{G}}{\sigma^{s-1}} && \text{def. of } t_0\\
&= \frac{\mathcal{G}}{\sigma^s}. && \sigma = \mu/(\vartheta-1)
\end{aligned}$$

Thus, $\Psi_w^s(t') > 0$, and we must have that $t' = t_0$ from the definition of t' and continuity of Ψ_w^s. As $\Psi^s(0) = 0$ by Lemma 8.26, this entails that $t_0 \neq 0$. Hence, $t' = t_0 = t_1 - \mathcal{G}/(\mu\sigma^{s-1})$. If $s > 1$, the minimality of s yields that $\Psi^s(t_0) \leq \mathcal{G}/\sigma^{s-1}$. We apply Lemma 8.25 to level s, node w, and time $t' = t_0$, yielding the contradiction

$$
\begin{aligned}
\frac{\mathcal{G}}{\sigma^s} + \varepsilon &= \Psi_w^s(t_1) && \text{(8.9)}\\
&\leq \Psi_w^s(t_0) + \vartheta(t_1 - t_0) - (L_w(t_1) - L_w(t_0)) && \text{Lemma 8.21}\\
&\leq (\vartheta - 1)(t_1 - t_0) && \text{Lemma 8.25}\\
&= \frac{\vartheta - 1}{\mu} \cdot \frac{\mathcal{G}}{\sigma^{s-1}} && t_0 = t_1 - \mathcal{G}/(\mu\sigma^{s-1})\\
&= \frac{\mathcal{G}}{\sigma^s}. && \sigma = \mu/(\vartheta - 1)
\end{aligned}
$$

Reaching a contradiction in all cases, we conclude that the statement of the theorem must indeed hold. □

The main result of our analysis readily follows.

Theorem 8.28. *Suppose that $\kappa \geq \delta$ and $H_v(0) - H_w(0) \leq \kappa$ for all edges $\{v, w\} \in E$. Then Algorithm 7 maintains a local skew of*

$$\mathcal{L} \leq 2\kappa \left\lceil \log_\sigma \frac{\mathcal{G}}{\kappa} \right\rceil,$$

where $\sigma := \mu/(\vartheta - 1)$.

Proof. We apply Lemma 8.27 for $s := \lceil \log_\sigma(\mathcal{G}/\kappa) \rceil$. For any $\{v, w\} \in E$ and any time t, we thus have that

$$
\begin{aligned}
L_v(t) - L_w(t) - (2s-1)\kappa &= L_v(t) - L_w(t) - (2s-1)\kappa \operatorname{dist}(v, w) && \{v, w\} \in E\\
&\leq \Psi^s(t) && \text{def. of } \Psi^s\\
&\leq \frac{\mathcal{G}}{\sigma^s} && \text{Lemma 8.27}\\
&\leq \kappa. && s \geq \log_\sigma(\mathcal{G}/\kappa)
\end{aligned}
$$

By exchanging the roles of v and w, we analogously obtain that $L_w(t) - L_v(t) - (2s-1)\kappa \leq \kappa$. Rearranging these inequalities, we conclude

$$\mathcal{L}(t) = \max_{\{v,w\} \in E} \{|L_v(t) - L_w(t)|\} \leq 2s\kappa = 2\kappa \lceil \log_\sigma(\mathcal{G}/\kappa) \rceil. \qquad \square$$

Bound on the Global Skew Theorem 8.28 bounds the local skew as function of the global skew. Since the theorem bounds the local skew for *any* algorithm implementing **FC** and **SC**, we have a lot of freedom regarding how to achieve a good global skew. For instance, we could follow the same strategy as in Algorithm 4 or Algorithm 5, but amortize clock jumps (i.e., run at rate $1 + \mu$ until the desired correction is accounted for) and give precedence to **ST**. Since **ST** prevents the logical clock from running fast only if a neighbor's clock is lagging behind even further, this is sufficient to maintain a small global skew.

Instead, we will bound the global skew solely based on **FC** and **SC**, establishing that *any* algorithm implementing these rules will perform well. Note, however, that more elaborate solutions like the one above can guarantee tighter bounds on the global skew.

Theorem 8.29. *Assume that $\kappa \geq \delta$ and $H_v(0) - H_w(0) \leq \kappa$ for all $\{v, w\} \in E$. Then Algorithm 7 satisfies $\mathcal{G} \leq (1 + 1/(\sigma - 1))\kappa D$, where $\sigma = \mu/(\vartheta - 1)$.*

Proof. Assume for contradiction that

$$\mathcal{G}(t_1) = \left(1 + \frac{1}{\sigma - 1}\right)\kappa D + \varepsilon = \frac{\sigma}{\sigma - 1} \cdot \kappa D + \varepsilon \tag{8.10}$$

for some (arbitrarily small) $\varepsilon > 0$ and a minimal time t_1. Such a time must exist, because $\mathcal{G}$ is continuous and

$$\begin{aligned}
\mathcal{G}(0) &= \max_{v,w \in V} \{L_v(0) - L_w(0)\} && \text{def. of } \mathcal{G}(0) \\
&\leq \max_{v,w \in V} \{L_v(0) - L_w(0) - \kappa \operatorname{dist}(v, w)\} + \kappa D && \operatorname{dist}(v,w) \leq D \\
&= \Psi^1(0) + \kappa D && \text{def. of } \Psi^1 \\
&= \kappa D. && \text{Lemma 8.26}
\end{aligned}$$

Set $t_0 := \max\{t_1 - \mathcal{G}(t_1)/\mu, 0\}$ and choose some w such that $\Psi^1_w(t_1) = \Psi^1(t_1)$. If $t_0 = 0$, it holds that

$$\begin{aligned}
\Psi^1_w(t_1) &\leq \Psi^1_w(0) + (\vartheta - 1)(t_1 - t_0) && \text{Corollary 8.22} \\
&= (\vartheta - 1)(t_1 - t_0). && (8.11) \quad \text{Lemma 8.26}
\end{aligned}$$

Otherwise, Lemma 8.25 is applicable, because $t_1 - t_0 = \mathcal{G}(t_1)/\mu > \mathcal{G}(t_0)/\mu$ by the minimality of t_1. Therefore, we get that

$$\begin{aligned}
\Psi^1_w(t_1) &\leq \Psi^1_w(t_0) - (L_w(t_1) - L_w(t_0)) + \vartheta(t_1 - t_0) && \text{Lemma 8.21} \\
&= (\vartheta - 1)(t_1 - t_0) && (8.12) \quad \text{Lemma 8.25}
\end{aligned}$$

as well. Thus, in both cases, we have that

$$
\begin{aligned}
\frac{\mathcal{G}(t_1)}{\sigma} &= \frac{\kappa D}{\sigma - 1} + \frac{\varepsilon}{\sigma} && \text{(8.10)}\\
&< \frac{\kappa D}{\sigma - 1} + \varepsilon && \varepsilon > 0,\ \sigma = \mu/(\vartheta - 1) > 1\\
&= \mathcal{G}(t_1) - \kappa D && \text{(8.10)}\\
&\le \max_{x,y \in V} \{L_x(t_1) - L_y(t_1) - \kappa \operatorname{dist}(x,y)\} && \operatorname{dist}(x,y) \le D\\
&= \Psi^1(t_1) && \text{def. of } \Psi^1\\
&= \Psi^1_w(t_1) && \text{choice of } w\\
&\le (\vartheta - 1)(t_1 - t_0) && \text{(8.11), (8.12)}\\
&\le \frac{(\vartheta - 1)\mathcal{G}(t_1)}{\mu} && \text{def. of } t_0\\
&= \frac{\mathcal{G}(t_1)}{\sigma}. && \sigma = \mu/(\vartheta - 1)
\end{aligned}
$$

This is a contradiction, implying that the claim of the theorem must hold true. □

Corollary 8.30. *Suppose that $\kappa \ge \delta$ and $H_v(0) - H_w(0) \le \kappa$ for all edges $\{v, w\} \in E$. Then Algorithm 7 maintains a local skew of*

$$\mathcal{L} \le 2\kappa \left\lceil \log_\sigma \frac{\sigma D}{\sigma - 1} \right\rceil,$$

where $\sigma := \mu/(\vartheta - 1)$.

8.5.4 Computing Clock Estimates

The remaining piece of the puzzle is how to generate the clock estimates $\tilde{L}^v_w(t)$. A simple strategy is for each node w to periodically broadcast its logical clock value to its neighbors. Each neighbor v then computes $\tilde{L}^v_w(t)$ using the known bounds on message delays, and increases $\tilde{L}^v_w$ at rate $\frac{dH_v}{dt}/\vartheta$ between messages from w. See Algorithm 8 for the pseudocode of this approach.

An upper bound on the error parameter δ can now be computed as a function of the period between broadcasts T, u, ϑ, and μ.

Lemma 8.31. *At times $t \ge T + d$, Algorithm 8 computes estimates satisfying (8.6) for*

$$\delta = \left(\vartheta(1 + \mu) - \frac{1}{\vartheta}\right)(T + u) + \vartheta(u + \mu d).$$

Assuming that $\mu \in O(1)$, this implies that $\delta = O(u + \mu(T + d))$.

Algorithm 8 Pseudocode for neighbors $v, w \in V$ keeping track of each other's logical clock based on regular updates. In order to generate all estimates needed for Algorithm 7, each node executes this code for each of its incident edges.

```
if v just woke up, i.e., t = 0 then
    h ← getH()
    ℓ̃_w ← getL()                              ▹ default to own clock value
end if
if getH() = kT for some k ∈ ℕ then
    send ⟨getL()⟩ to w
end if
if received ⟨ℓ⟩ from w then
    ℓ̃_w ← ℓ + d − u                           ▹ take message delay into account
    h ← getH()
end if
procedure getL(w)                              ▹ returns L̃_v^w(t)
    return ℓ̃_w + (getH() − h)/ϑ               ▹ own clock might be faster than w's
end procedure
```

Proof. Fix $\{v, w\} \in E$ and consider time $t \geq T + d$. As messages are sent every T local time, $\frac{dH_w}{dt} \geq 1$, and all messages are received within d time, v received a message from w by time $T + d$. Denote by t_s the time it was sent and by $t_r \leq t$ the time it was received. We have that $t_s + d - u \leq t_r \leq t_s + d$. Hence,

$$
\begin{aligned}
\tilde{L}_w(t) &\leq \tilde{L}_w(t_r) + \frac{H_v(t) - H_v(t_r)}{\vartheta} && \tfrac{d\tilde{L}_w}{dt} \geq \tfrac{1}{\vartheta} \cdot \tfrac{dH_v}{dt} \\
&\leq \tilde{L}_w(t_r) + t - t_r && \tfrac{dH_v}{dt} \leq \vartheta \\
&= L_w(t_s) + d - u + t - t_r && d - u \text{ added on reception} \\
&\leq L_w(t_s) + t_r - t_s + t - t_r && d - u \leq t_r - t_s \\
&= L_w(t_s) + t - t_s \\
&\leq L_w(t_s) + H_w(t) - H_w(t_s) && \tfrac{dH_w}{dt} \geq 1 \\
&\leq L_w(t_s) + L_w(t) - L_w(t_s) = L_w(t) \qquad (8.13) && \tfrac{dL_w}{dt} \geq \tfrac{dH_w}{dt}
\end{aligned}
$$

and

$$
\begin{aligned}
L_w(t) &= L_w(t_s) + L_w(t) - L_w(t_s) && \text{adding } 0\\
&\leq L_w(t_s) + (1+\mu)(H_w(t) - H_w(t_s)) && \tfrac{dL}{dt} \leq \tfrac{(1+\mu)dH}{dt}\\
&\leq L_w(t_s) + (1+\mu)\vartheta(t - t_s) && \tfrac{dH_w}{dt} \leq \vartheta\\
&= \tilde{L}_w(t_r) - (d-u) + (1+\mu)\vartheta(t - t_s) && d-u \text{ added on reception}\\
&= \tilde{L}_w(t_r) - (d-u) + (1+\mu)\vartheta(t - t_r + t_r - t_s) && \text{adding } 0\\
&= \tilde{L}_w(t_r) + (1+\mu)\vartheta(t - t_r) + \vartheta(u + \mu d) && t_r - t_s \leq d\\
&= \tilde{L}_w(t) - (\tilde{L}_w(t) - \tilde{L}_w(t_r)) + (1+\mu)\vartheta(t - t_r) + \vartheta(u + \mu d) && \text{adding } 0\\
&\leq \tilde{L}_w(t) - \frac{H_w(t) - H_w(t_r)}{\vartheta} + (1+\mu)\vartheta(t - t_r) + \vartheta(u + \mu d) && \tfrac{dL_w}{dt} \geq \tfrac{dH_w}{dt}\\
&\leq \tilde{L}_w(t) - \frac{t - t_r}{\vartheta} + (1+\mu)\vartheta(t - t_r) + \vartheta(u + \mu d) && \tfrac{dH_w}{dt} \geq 1\\
&= \tilde{L}_w(t) + \left(\vartheta(1+\mu) - \frac{1}{\vartheta}\right)(t - t_r) + \vartheta(u + \mu d). && (8.14)
\end{aligned}
$$

In order to bound $t - t_r$, recall that t_s is the time the latest message from w that v received was sent. Hence, if t'_s is the next time after t_s when w sends a message, we have that $t - t'_s < d$. We get that

$$
\begin{aligned}
t - t_r &= t - t'_s + t'_s - t_s - (t_r - t_s) && \text{adding } 0\\
&< t'_s - t_s + u && t - t'_s < d,\ t_r - t_s \geq d - u\\
&\leq H_w(t'_s) - H_w(t_s) + u && \tfrac{dH_w}{dt} \geq 1\\
&= T + u. && \text{mess. sent every } T \text{ local time} \quad (8.15)
\end{aligned}
$$

We conclude that

$$
\begin{aligned}
L_w(t) &\geq \tilde{L}_w(t) && (8.13)\\
&\geq \tilde{L}_w(t) - \left(\vartheta(1+\mu) - \frac{1}{\vartheta}\right)(t - t_r) + \vartheta(u + \mu d) && (8.14)\\
&> \tilde{L}_w(t) - \left(\vartheta(1+\mu) - \frac{1}{\vartheta}\right)(T + u) + \vartheta(u + \mu d) && (8.15)\\
&= \tilde{L}_w(t) - \delta,
\end{aligned}
$$

as claimed.

To establish the asymptotic bound on δ, note that $\vartheta = 1 + \vartheta - 1 < 1 + \mu$. Thus,

$$\begin{aligned}
\vartheta(1+\mu) - \frac{1}{\vartheta} &< (1+\mu)^2 - \frac{1}{1+\mu} && \vartheta < 1+\mu \\
&< (1+\mu)^3 - 1 && 1+\mu > 1 \\
&= O(\mu), \quad (8.16) && \mu = O(1)
\end{aligned}$$

leading to

$$\begin{aligned}
\left(\vartheta(1+\mu) - \frac{1}{\vartheta}\right)(T+u) + \vartheta(u+\mu d) &= O(\mu(T+u) + (1+\mu)(u+\mu d) && (8.16) \\
&= O(\mu(T+u) + u + \mu d) && \mu = O(1) \\
&= O(u + \mu(T+d)). \quad \square && u \le d
\end{aligned}$$

Finally, we can put all pieces together to arrive at Corollary 8.32.

Corollary 8.32. *Suppose that*

- $H := \max_{\{v,w\}\in E}\{H_v(0) - H_w(0)\} \in O(u)$,
- *Algorithm 8 with $T = d$ is used to compute clock estimates and is initialized at all nodes by time $-2d$,*
- $2(\vartheta - 1) \le \mu = O(u/d)$, *and*
- $\kappa = \max\{H, \delta\}$, *where δ is as in Lemma 8.31.*

Then Algorithm 7 guarantees that

- $\frac{dH_v}{dt}(t) \le \frac{dL_v}{dt}(t) \le (1+\mu)\frac{dH_v}{dt}(t)$ *for all nodes v and times t,*
- $\mathcal{G} = O(D)$, *and*
- $\mathcal{L} = O(u \log_\sigma D)$, *where $\sigma = \mu/(\vartheta - 1)$.*

Proof. The bounds on the logical clock rates are immediate from Algorithm 7. By Lemma 8.31, the choice of $T = d$, and the assumption that Algorithm 8 is initialized at all nodes by time $-2d$, at times $t \ge 0$ all nodes maintain estimates of their neighbors' logical clocks satisfying (8.6) for the δ specified in the lemma. Because $\mu = O(u/d)$ and $u \le d$, $u \in O(1)$. Thus,

$$\begin{aligned}
\delta &= O(u + \mu(T+d)) && \mu = O(1), \text{ Lemma 8.31} \\
&= O(u + \mu d) && T = d \\
&= O(u), && \mu = O(u/d)
\end{aligned}$$

implying that $\kappa = O(u)$. Moreover, $\sigma = \mu/(\vartheta - 1) \ge 2$. The choice of κ satisfies the prerequisites of Theorem 8.29 and Corollary 8.30. Thus, we get that

$$\begin{aligned} \mathcal{G} &\leq \left(1 + \frac{1}{\sigma - 1}\right)\kappa D && \text{Theorem 8.29} \\ &= O(uD) && \sigma \geq 2,\ \kappa = O(u) \end{aligned}$$

and

$$\begin{aligned} \mathcal{L} &\leq 2\kappa\left\lceil \log_\sigma \frac{\sigma D}{\sigma - 1} \right\rceil && \text{Corollary 8.30} \\ &\leq 2\kappa\left\lceil \log_\sigma \frac{\sigma}{\sigma - 1} + \log_\sigma D \right\rceil && \log(ab) = \log a + \log b \\ &\leq 2\kappa\lceil 1 + \log_\sigma D \rceil && \sigma \geq 2 \\ &= O(u \log_\sigma D). && \kappa = O(u) \end{aligned}$$

□

A Technical Lemma

If the maximum of a finite set of functions satisfies that whenever one of the functions attains the maximum, its derivative is bounded by r, then the growth of the maximum is also bounded by r. This intuitive statement is surprisingly hard to prove.

Lemma 8.33. *For $k \in \mathbb{N}$, let $\mathcal{F} = \{f_i \,|\, i \in [k]\}$, where each $f_i \colon [t_0, t_1] \to \mathbb{R}$ is differentiable, and $[t_0, t_1] \subset \mathbb{R}$. Define $F \colon [t_0, t_1] \to \mathbb{R}$ by $F(t) := \max_{i \in [k]} \{f_i(t)\}$. Suppose $\mathcal{F}$ has the property that for every i and t, if $f_i(t) = F(t)$, then $\frac{df_i}{dt}(t) \leq r$. Then for all $t \in [t_0, t_1]$, we have $F(t) \leq F(t_0) + r(t - t_0)$.*

Proof. We prove the stronger claim that for all a, b satisfying $t_0 \leq a < b \leq t_1$, we have

$$\frac{F(b) - F(a)}{b - a} \leq r. \tag{8.17}$$

To this end, suppose to the contrary that there exist $a_0 < b_0$ satisfying $(F(b_0) - F(a_0))/(b_0 - a_0) \geq r + \varepsilon$ for some $\varepsilon > 0$. We define a sequence of nested intervals $[a_0, b_0] \supset [a_1, b_1] \supset \cdots$ as follows. Given $[a_j, b_j]$, let $c_j = (b_j + a_j)/2$ be the midpoint of a_j and b_j. Observe that

$$\frac{F(b_j) - F(a_j)}{b_j - a_j} = \frac{1}{2}\frac{F(b_j) - F(c_j)}{b_j - c_j} + \frac{1}{2}\frac{F(c_j) - F(a_j)}{c_j - a_j} \geq r + \varepsilon,$$

so that

$$\frac{F(b_j) - F(c_j)}{b_j - c_j} \geq r + \varepsilon \quad \text{or} \quad \frac{F(c_j) - F(a_j)}{c_j - a_j} \geq r + \varepsilon.$$

If the first inequality holds, define $a_{j+1} = c_j$, $b_{j+1} = b_j$, and otherwise define $a_{j+1} = a_j$, $b_j = c_j$. From the construction of the sequence, it is clear that for all j we have

$$\frac{F(b_j) - F(a_j)}{b_j - a_j} \geq r + \varepsilon. \tag{8.18}$$

Observe that the sequences $\{a_j\}_{j=0}^{\infty}$ and $\{b_j\}_{j=0}^{\infty}$ ar both bounded and monotonic, hence convergent. Further, since $b_j - a_j = \frac{1}{2^j}(b_0 - a_0)$, the two sequences share the same limit.

Define

$$c := \lim_{j\to\infty} a_j = \lim_{j\to\infty} b_j,$$

and let $f \in \mathcal{F}$ be a function satisfying $f(c) = F(c)$. By the hypothesis of the lemma, we have $f'(c) \leq r$, so that

$$\lim_{h\to 0} \frac{f(c+h) - f(h)}{h} \leq r.$$

Therefore, there exists some $h > 0$ such that for all $t \in [c-h, c+h]$, $t \neq c$, we have

$$\frac{f(t) - f(c)}{t - c} \leq r + \frac{1}{2}\varepsilon.$$

Further, from the definition of c, there exists $N \in \mathbb{N}$ such that for all $j \geq N$, we have $a_j, b_j \in [c-h, c+h]$. In particular this implies that for all sufficiently large j, we have

$$\frac{f(c) - f(a_j)}{c - a_j} \leq r + \frac{1}{2}\varepsilon, \tag{8.19}$$

$$\frac{f(b_j) - f(c)}{b_j - c} \leq r + \frac{1}{2}\varepsilon. \tag{8.20}$$

Since $f(a_j) \leq F(a_j)$ and $f(c) = F(c)$, (8.19) implies that for all $j \geq N$,

$$\frac{F(c) - F(a_j)}{c - a_j} \leq r + \frac{1}{2}\varepsilon.$$

However, this expression combined with with (8.18) implies that for all $j \geq N$

$$\frac{F(b_j) - F(c)}{b_j - c} \geq r + \varepsilon. \tag{8.21}$$

Since $F(c) = f(c)$, the previous expression together with (8.20) implies that for all $j \geq N$ we have $f(b_j) < F(b_j)$.

For each $j \geq N$, let $g_j \in \mathcal{F}$ be a function such that $g_j(b_j) = F(b_j)$. Since $\mathcal{F}$ is finite, there exists some $g \in \mathcal{F}$ such that $g = g_j$ for infinitely many values j. Let $j_0 < j_1 < \cdots$ be the subsequence such that $g = g_{j_k}$ for all $k \in \mathbb{N}$. Then for all j_k, we have $F(b_{j_k}) = g(b_{j_k})$. Further, since F and g are continuous, we have

$$g(c) = \lim_{k\to\infty} g(b_{j_k}) = \lim_{k\to\infty} F(b_{j_k}) = F(c) = f(c).$$

By (8.21), we therefore have that for all k

$$\frac{g(b_{j_k}) - g(c)}{b_{j_k} - c} = \frac{F(b_j) - F(c)}{b_j - c} \geq r + \varepsilon.$$

However, this final expression contradicts the assumption that $g'(c) \leq r$. Therefore, (8.17) holds, as desired. □