Lecture 3

Upper Bound on the Local Skew

In Chapter 1, we proved tight upper and lower bounds of $\Theta(D)$ for the global skew of any clock synchronization algorithm. However, the algorithms achieving optimal global skew had the undesireable feature that the maximal global skew could be attained between any pair of nodes in the network — even adjacent nodes. In Chapter 2, we introduced the local skew — the maximum skew between any pair of *adjacent* nodes — and learned that it must be at least logarithmic in D. In this chapter, we address the question of whether the trivial $\mathcal{O}(D)$ bound on the local skew given by the best worst-case bound on the global skew can be improved. In particular, we present a GCS algorithm proving the lower bound to be asymptotically tight.

3.1 GCS Algorithm

The high-level strategy of the algorithm is as follows. As with the naive algorithms from the previous chapter, 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 $h_v(t)$. In fast mode, v will increase its logical clock at rate $(1 + \mu)h_v(t)$. The parameter μ will be chosen large enough for nodes whose logical clocks are behind to be able to catch up to other nodes. 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 3.1 (**FC**: Fast Mode Condition)**.** *We say that a node $v \in V$ satisfies the* fast mode condition (**FC**) *at time $t \in \mathbb{R}_0^+$ if there exists $s \in \mathbb{N}$ such that:*

FC-1 $\exists x \in N_v \colon L_x(t) - L_v(t) \geq 2s\kappa$,

FC-2 $\forall y \in N_v \colon L_v(t) - L_y(t) \leq 2s\kappa$.

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 $x \in N_v$ satisfying **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 3.2 (**SC**: Slow Mode Condition)**.** *We say that a node $v \in V$ satisfies the* slow mode condition *(or* **SC***) at time $t \in \mathbb{R}_0^+$ if there exists $s \in \mathbb{N}$ such that:*

SC-1 $\exists x \in N_v \colon L_v(t) - L_x(t) \geq (2s-1)\kappa$,

SC-2 $\forall y \in N_v \colon L_y(t) - L_v(t) \leq (2s-1)\kappa$.

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

$$L_w(t) \geq \tilde{L}_w^v(t) \geq L_w(t) - \delta. \tag{3.1}$$

For convenience we occasionally 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 $\tilde{L}_x^v(t) - L_v(t) < 2s\kappa$, so that v does see that **FC-1** is satisfied. Since $\tilde{L}_w^v(t) \geq L_w(t) - \delta$, v *might* satisfy **FC-1** if $\tilde{L}_w^v(t) - L_v(t) \geq 2s\kappa - \delta$. 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 3.3 (**FT**: Fast Mode Trigger)**.** *We say that $v \in V$ satisfies the* fast mode trigger (**FT**) *at time $t \in \mathbb{R}_0^+$ if there exists an integer $s \in \mathbb{N}$ such that:*

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

FT-2 $\forall y \in N_v \colon L_v(t) - \tilde{L}_y(t) < 2s\kappa + \delta.$

Definition 3.4 (**ST**: Slow Mode Trigger)**.** *We say that a node $v \in V$ satisfies the* slow mode trigger *(or* **ST***) at time $t \in \mathbb{R}_0^+$ if there exists $s \in \mathbb{N}$ such that:*

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

ST-2 $\forall y \in N_v \colon \tilde{L}_y(t) - L_v(t) \leq (2s-1)\kappa.$

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, $w \in N_v$, and times t we have $\tilde{L}_w^v(t) \leq L_w(t)$. 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 3.5. *Suppose $\kappa > \delta$. Then no node $v \in V$ can simultaneously satisfy* **FT** *and* **ST**.

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

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

so **ST-1**is not satisfied for s'. If $s' \leq s$, then there is some $x \in N_v$ satisfying

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

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

Lemma 3.6. *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 (3.1), there is some $s \in \mathbb{N}$ such that

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

and

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

Thus **FT** holds. Similarly, if **SC** holds, (3.1) yields

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

and

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

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

We now describe the GCS algorithm. To focus on the key ideas of the analysis, we make another simplifying assumption: Instead of analyzing the global skew, we assume that it is bounded by some parameter $\mathcal{G}$. You will address the issue of giving an explicit expression for $\mathcal{G}$ in an exercise. If we ignore the issue of bounded $\mathcal{G}$, the GCS algorithm is extremely simple. Each node v initializes its logical clock to its hardware clock value. It continuously checks if the fast (resp. slow) mode trigger is satisfied. If so, it increases its logical clock at a rate of $(1+\mu)h_v(t)$ (resp. $h_v(t)$). Pseudocode is presented in Algorithm 3.1. Despite the algorithms simplicity, its analysis (presented in the following section) is rather delicate.

Algorithm 3.1: GCS algorithm

```
1 $L_v(0) := H_v(0)$
2 $r := 1$
3 at all times $t$ do the following
4 if FT then
5     $r := 1 + \mu$                    // $v$ is in fast mode
6 if ST then
7     $r := 1$                          // $v$ is in slow mode
8 increase $L_v$ at rate $r \cdot h_v(t)$
```

Remarks:

- When neither **FT** nor **ST** are satisfied, the logical clock may run at any speed from the range $[h_v(t), (1+\mu)h_v(t)]$. Indeed,

- In order for the algorithm to be implementable, we required $\kappa > \delta$. While δ is generally determined by the physical characteristics of the network (e.g., d, u, ϑ, etc.), the value of κ has significance in our analysis of the algorithm. In general, smaller values of κ will incur smaller (worst-case) local skews. The tradeoff is that as κ becomes closer to δ, nodes may switch between fast and slow modes arbitrarily fast. Thus in practice, it may be desireable to keep κ bounded away from δ.

- For technical reasons, we will assume that logical clocks are differentiable. Thus, $l_v := \frac{d}{dt}L_v$ exists and is between 1 and $\vartheta(1+\mu)$ at all times. It is possible to prove the guarantees of the algorithm without this assumption, but the assumption of differentiability simplifies our analysis.
- Even with the differentiability assumption, we still need Lemma 4.1. This is not a mathematics lecture, but as we couldn't find any suitable reference, the lemma and a proof is given in the appendix.

3.2 Analysis of the GCS Algorithm

We now show that the GCS algorithm (Algorithm 3.1) indeed achieves a small local skew, which is expressed by the following theorem.

Theorem 3.7. *For every network G and every execution $\mathcal{E}$ in which $H_v(0) - H_w(0) \leq \kappa$ for all edges $\{v,w\} \in E$, the GCS algorithm achieves a gradient skew of $\mathcal{L} \leq 2\kappa\lceil\log_\sigma \mathcal{G}/\kappa\rceil$, where $\sigma := \mu/(\vartheta-1)$.*

In order to prove Theorem 3.7, 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 (in particular edges) can only have a(n average) skew that is logarithmic in the network diameter.

Leading Nodes

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

Definition 3.8 (Ψ and Leading Nodes)**.** *For each $v \in V$, $s \in \mathbb{N}$, and $t \in \mathbb{R}_0^+$, we define*

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

where $d(v,w)$ 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*

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

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 $s \in \mathbb{N}$ and all times t. In particular, Theorem 3.7 follows 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 3.9 (Leading Lemma)**.** *Suppose $w \in V$ is a leading node at time t. Then w satisfies* **SC** *and* **ST**.

Proof. By Lemma 3.6, 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 $s \in \mathbb{N}$ and $v \in V$ satisfying

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

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

$$L_w(t) - L_v(t) - (2s-1)\kappa d(v,w) = \Psi_v^s(t) \geq L_y(t) - L_v(t) - (2s-1)\kappa d(y,w)\,.$$

Rearranging this yields

$$L_w(t) - L_y(t) \geq (2s-1)\delta(d(v,w) - d(y,w))\,.$$

In particular, for any $y \in N_v$, $d(v,w) \geq d(y,w) - 1$ and hence

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

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

Now consider $x \in N_v$ so that $d(x,w) = d(v,w) - 1$. 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 3.6, 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.

Lemma 3.10 (Wait-up Lemma)**.** *Suppose $w \in V$ satisfies $\Psi_w^s(t) > 0$ for all $t \in (t_0, t_1]$. Then*

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

Proof. Fix $w \in V$, $s \in \mathbb{N}$ 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 $f_v(t) = L_v(t) - (2s-1)\delta d(v,w)$. Observe that

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

Moreover, for any v satisfying $f_v(t) = L_w(t) + \Psi_w^s(t)$, we have that $L_v(t) - L_w(t) - (2s-1)\kappa d(v,w) = \Psi_w^s(t) > 0$. Thus, Lemma 3.9 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 $\frac{d}{dt} f_v(t) \leq \vartheta$ for any $v \in V$ and time $t \in (t_0, t_1]$ satisfying $f_v(t) = \max_{x \in V}\{f_x(t)\}$. By Lemma 4.1, it follows that $\max_{v \in V}\{f_v(t)\}$ grows at most at rate ϑ:

$$\max_{v \in V}\{f_v(t_1)\} \leq \max_{v \in V}\{f_v(t_0)\} + \vartheta(t_1 - t_0)\,.$$

We conclude that

$$\begin{aligned}\Psi_w^s(t_1) - \Psi_w^s(t_0) &= \max_{v \in V}\{f_v(t_1)\} - L_w(t_1) - (\max_{v \in V}\{f_v(t_0)\} - L_w(t_0)) \\ &\leq -(L_w(t_1) - L_w(t_0)) + \vartheta \cdot (t_1 - t_0)\,,\end{aligned}$$

which gives the desired result. □

Trailing Nodes

As $L_w(t_1) - L_w(t_0) \geq t_1 - t_0$ at all times, Lemma 3.13 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 3.10, where now we need to exploit the fast mode condition **FC**.

Definition 3.11 (Trailing Nodes)**.** *We say that $w \in V$ is a* trailing node *at time t, if there exists $s \in \mathbb{N}$ and a node $v \in V$ such that*

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

Lemma 3.12 (Trailing Lemma)**.** *Suppose $w \in V$ is a trailing node at time t. Then w satisfies* **FC** *and* **FT***.*

Proof. By Lemma 3.6, if w satisfies **FC**, then w also satisfies **FT**. Thus, it suffices to prove that w satisfies **FC**. Let s and v satisfy

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

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

$$L_v(t) - L_w(t) - 2s\kappa d(v,w) \geq L_v(t) - L_y(t) - 2s\kappa d(v,y).$$

Thus for all neighbors $y \in N_w$,

$$L_y(t) - L_w(t) + 2s\delta(d(v,y) - d(v,w)) \geq 0\,.$$

It follows that

$$\forall y \in N_v \colon L_w(t) - L_y(t) \leq 2s\kappa\,,$$

i.e., **FC-2** holds. As $v \neq w$, there is some node $x \in N_v$ with $d(v,x) = d(v,w) - 1$. We obtain that

$$\exists x \in N_v \colon L_y(t) - L_w(t) \geq 2s\kappa\,,$$

showing **FC-1**. □

Using Lemma 3.12, we can show that if $\Psi^s_w(t_0) > 0$, 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 3.13 (Catch-up Lemma)**.** *Let $s \in \mathbb{N}$ and t_0, t_1 be times. Suppose that*

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

Then, for any $w \in V$,

$$L_w(t_1) - L_w(t_0) \geq t_1 - t_0 + \Psi^s_w(t_0)\,.$$

Proof. Choose $v \in V$ such that

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

Define $f_x(t) \coloneqq L_v(t_0) + (t - t_0) - L_x(t) - (2s-2)\kappa d(v,x)$ for $x \in V$ and observe that $\Psi_w^s(t_0) \leq f_w(t_0)$. Hence, if $\max_{x\in V}\{f_x(t)\} \leq 0$ for some $t \in [t_0, t_1]$, then

$$\begin{aligned} L_w(t_1) - L_w(t) - (t_1 - t) &\geq 0 \geq f_w(t) \\ &= L_v(t_0) + (t - t_0) - L_w(t) - (2s-2)\kappa d(v,x) \\ &= f_w(t_0) + (t - t_0) - (L_w(t) - L_w(t_0)) \\ &\geq \Psi_w^s(t_0) + (t - t_0) - (L_w(t) - L_w(t_0))\,, \end{aligned}$$

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

$$\begin{aligned} &\max_{x\in V}\{L_v(t) - L_x(t) - (2s-2)\delta d(v,x)\} \\ &= L_v(t) - L_v(t_0) - (t - t_0) + \max_{x\in V}\{f_x(t)\} \\ &= L_v(t) - L_v(t_0) - (t - t_0) + f_y(t) \\ &= L_v(t) - L_y(t) - (2s-2)\delta d(v,y) \end{aligned}$$

and

$$L_v(t) - L_v(t_0) - (t - t_0) + \max_{x\in V}\{f_x(t)\} > L_v(t) - L_v(t_0) - (t - t_0) \geq 0\,.$$

Thus, by Lemma 3.12 y is in fast mode. As logical clocks are (assumed to be) differentiable, we obtain $\frac{d}{dt} f_y(t) = 1 - l_y(t) \leq -\mu$.

Now assume for contradiction that $\max_{x\in V}\{f_x(t)\} > 0$ for all $t \in [t_0, t_1]$. Then, applying Lemma 4.1 again, we conclude that

$$\max_{x\in V}\{f_x(t_0)\} > -(\max_{x\in V}\{f_x(t_1)\} - \max_{x\in V}\{f_x(t_0)\}) \geq \mu(t_1 - t_0)\,.$$

If $s = 1$, $\mu(t_1 - t_0) \geq \mathcal{G}$, contradicting the definition of $\mathcal{G}$:

$$f_x(t_0) = L_v(t_0) - L_x(t_0) \leq \mathcal{G}$$

for all $x \in V$. If $s > 1$, then $\mu(t_1 - t_0) \geq \Psi^{s-1}(t_0)$. However, we have that

$$f_x(t_0) \leq L_v(t_0) - L_x(t_0) - (2s-3)\delta d(v,x) \leq \Psi^{s-1}(t_0)$$

for all $x \in V$. As this is a contradiction as well, the claim of the lemma follows. □

Putting Things Together

Theorem 3.14. *Assume that $H_v(0) - H_w(0) \leq \kappa$ for all $\{v,w\} \in E$. Then, for all $s \in \mathbb{N}$, Algorithm 3.1 guarantees $\Psi^s(t) \leq \mathcal{G}/\sigma^s$, where $\sigma = \mu/(\vartheta - 1)$.*

Proof. Suppose for contradiction that the statement of the theorem is false. Let $s \in \mathbb{N}$ 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 $\varepsilon > 0$. Thus, there is some $w \in V$ such that

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

Set $t_0 := \max\{t - \mathcal{G}/(\mu\sigma^{s-1}), 0\}$. 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). Thus, we can apply Lemma 3.10 to this interval, yielding that

$$\Psi^s_w(t_1) \le \Psi^s_w(t') + \vartheta(t_1 - t') - (L_w(t_1) - L_w(t')) \le \Psi^s_w(t') + (\vartheta - 1)(t_1 - t') \,.$$

$\Psi^s_w(t')$ cannot be 0, as otherwise

$$\Psi^s_w(t_1) \le (\vartheta - 1)(t_1 - t') \le \frac{\vartheta - 1}{\mu} \cdot \frac{\mathcal{G}}{\sigma^{s-1}} = \frac{\mathcal{G}}{\sigma^s} \,,$$

contradicting $\Psi^s_w(t_1) = \mathcal{G}/\sigma^s + \varepsilon$.

On the other hand, if $\Psi^s_w(t') > 0$, we must have $t' = t_0$ from the definition of t' and continuity of Ψ^s_w. Moreover, $t_0 \neq 0$ because

$$\begin{aligned}
&\max_{v,w \in V}\{L_v(0) - L_w(0) - (2s-1)\kappa d(v,w)\} \\
=\; &\max_{v,w \in V}\{H_v(0) - H_w(0) - (2s-1)\kappa d(v,w)\} \\
\le\; &\max_{v,w \in V}\{H_v(0) - H_w(0) - \kappa d(v,w)\} \le 0 \,,
\end{aligned}$$

as $H_v(0) - H_w(0) \le \kappa$ for all neighbors v, w by assumption. Hence, $t' = t_0 = t_1 - \mathcal{G}/(\mu\sigma^{s-1})$. If $s > 1$, the minimality of s yields that $\Psi^s(t_0) \le \mathcal{G}/\sigma^{s-1}$. We apply Lemma 3.13 to level s, node w, and time $t' = t_0$, yielding that

$$\Psi^s_w(t_1) \le \Psi^s_w(t_0) + \vartheta(t_1 - t_0) - (L_w(t_1) - L_w(t_0)) \le (\vartheta - 1)(t_1 - t_0) \le \frac{\mathcal{G}}{\sigma^s} \,,$$

again contradicting $\Psi^s_w(t_1) = \mathcal{G}/\sigma^s + \varepsilon$. Reaching a contradiction in all cases, we conclude that the statement of the theorem must indeed hold. □

Our main result, Theorem 3.7, is now immediate.

Proof of Theorem 3.7. We apply Theorem 3.14 and consider $s := \lceil \log_\sigma(\mathcal{G}/\kappa) \rceil$. For any $\{v, w\} \in E$ and any time t, we thus have that

$$L_v(t) - L_w(t) - (2s-1)\kappa = L_v(t) - L_w(t) - (2s-1)\kappa d(v,w) \le \Psi^s(t) \le \frac{\mathcal{G}}{\sigma^s} \le \kappa \,.$$

Rearranging this and exchanging the roles of v and w, we obtain

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

Remarks:

- Assuming a number of reasonable things and that $T \in \mathcal{O}(d)$ (i.e., message frequency is not the bottleneck in determining estimates), an asymptotically optimal choice of μ you will compute in the exercises yields a skew of roughly $2u \log_\sigma D$ for our GCS algorithm. Thus, the lower bounds from the Lecture 3 shows that the algorithm is optimal up to a factor of roughly 2, provided $\sigma \gg 1$ and $(\vartheta - 1)d \ll u$. Dropping that $\sigma \gg 1$, we still get optimality up to a constant factor.

- What if $(\vartheta - 1)d$ is comparable to u or even larger? In the exercises, you show how to generate a better "logical hardware clock" in this case by bouncing messages back and forth between nodes. Using this idea (with some modifications), one *could*, up to an additive $\mathcal{O}((\vartheta - 1)d)$, eliminate the dependence of the upper bound on $(\vartheta - 1)d$.

What to Take Home

- A very simple algorithm achieves a surprisingly good local skew, even if clocks must advance at all times.

- The base of the logarithm in the bound is typically large. A cheap quartz oscillator guarantees $\vartheta - 1 \leq 10^{-5}$. Thus, even if we want that $\mu \ll 1$, the base of the logarithmic term can be made quite large.

- The algorithmic idea is surprisingly versatile. It works if δ is different for each link, and with some modifications (to algorithm and analysis), adversarial changes in the graph can be handled.

Bibliographic Notes

The first non-trivial upper bound for the GCS problem was provided by Locher and Wattenhofer [LW06]. Their *blocking algorithm* bounds the local skew by $\mathcal{O}(\sqrt{\delta D})$. The first logarithmic bound on the local skew was given in [LLW08] and soon after improved to the algorithm presented here [LLW10]. The elegant way of describing the algorithm in terms of the fast and slow modes and conditions is due to Kuhn and Oshman [KO09].

The underlying idea of the GCS algorithm presented here turns out to be surprisingly robust and versatile. Essentially the same algorithm works for different uncertainties on the edges [KO09]. With a suitable method of carefully incorporating newly appearing edges, it can handle dynamic graphs [KLLO10] (this problem is introduced in [KLO11]), in the sense that edges that were continuously present for sufficiently long satisfy the respective guarantee on the skew between their endpoints. Recently, the approach has been independently discovered (twice!) for solving load balancing tasks that arise in certain packet routing problems [DLNO17, PR17].

Bibliography

[DLNO17] Stefan Dobrev, Manuel Lafond, Lata Narayanan, and Jaroslav Opatrny. Optimal local buffer management for information gathering with adversarial traffic. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2017, Washington DC, USA, July 24-26, 2017*, pages 265–274, 2017.

[KLLO10] Fabian Kuhn, Christoph Lenzen, Thomas Locher, and Rotem Oshman. Optimal Gradient Clock Synchronization in Dynamic Networks. *CoRR*, abs/1005.2894, 2010.

[KLO11] Fabian Kuhn, Thomas Locher, and Rotem Oshman. Gradient Clock Synchronization in Dynamic Networks. *Theory Comput. Syst.*, 49(4):781–816, 2011.

[KO09] Fabian Kuhn and Rotem Oshman. Gradient Clock Synchronization Using Reference Broadcasts. In *Proc. 13th Conference on Principles of Distributed Systems (OPODIS)*, pages 204–218, 2009.

[LLW08] Christoph Lenzen, Thomas Locher, and Roger Wattenhofer. Clock Synchronization with Bounded Global and Local Skew. In *Proc. 49th Symposium on Foundations of Computer Science (FOCS)*, pages 509–518, 2008.

[LLW10] Christoph Lenzen, Thomas Locher, and Roger Wattenhofer. Tight Bounds for Clock Synchronization. *J. ACM*, 57(2):8:1–8:42, 2010.

[LW06] Thomas Locher and Roger Wattenhofer. Oblivious Gradient Clock Synchronization. In *Proc. 20th Symposium on Distributed Computing (DISC)*, pages 520–533, 2006.

[PR17] Boaz Patt-Shamir and Will Rosenbaum. The space requirement of local forwarding on acyclic networks. In *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 13–22, 2017.