Exercise 1: Global vs. Local

Task 1: Even more Globally Optimal (60 points)

In the lecture, we proved a lower bound of $\Omega(uD)$ for the global skew of any algorithm, while the uppper bound achieved by the max algorithm is $\mathcal{O}(((\vartheta-1)d+u)D)$. Thus, the upper bound may be significantly larger than our lower bound in the regime where $(\vartheta-1)d \gg u$. In this exercise, you will show that an upper bound of $\mathcal{O}(uD+(\vartheta-1)d)$ is always achievable.

(a) Let $v \in V$ be a node and $w \in N_v$ an arbitrary neighbor. Consider the following process: at time $t=0$, v sends a message to w. When w receives v's message, w immediately replies with a message. When v receives w's message, v again responds immediately and so on. Thus, for any exection, this process defines a sequence of times $0=t_0<t_1<t_2<\cdots$ when v receives a message from w. Define a "hardware clock" $\widehat{H}_v$ that (discontinuously) adjusts its value at times $t_1, t_2, \dots$ and increases at the same rate as H_v at all other times such that for all times $t<t'$, $\widehat{H}_v$ satisfies

$$\widehat{H}_v(t')-\widehat{H}_v(t) \le \left(1+\frac{u}{d}\right)(t'-t)+c(u+(\vartheta-1)d) \quad \text{and} \quad H_v(0)+t \le \widehat{H}_v(t).$$

for some suitable constant $c>0$. (25 points) *Hint: What do you know about $t_{i+1}-t_i$?*

(b) Consider the "Min Algorithm," in which nodes reduce their logical clock value to $L+d$ when receiving a message $\langle L\rangle$, but use $\widehat{H}_v$ as "hardware" clock at v. (The messages sent are specified in c)). Define a "virtual clock" $L_{\min}$ that lower bounds all actual clock values, but also takes into account messages that are in transit, which might cause nodes to set their clocks back. This clock should satisfy that it increases at least at rate 1. Conclude that the algorithm guarantees amortized 1-progress. (10 points)

(c) Bound the global skew of the algorithm. We are not concerned with the number of sent messages, so nodes will send a message immediately after receiving one, as well as send one every d local time (w.r.t. H_v). (Remark: If it helps with notation, feel free to show the statement only on a graph that is a simple path.) (25 points)

Task 2: The Least Possible Locally Optimal (40 points)

The *local skew* is defined in a similar way as the global skew except that we consider the logical clock difference between the neighboring nodes only:

$$\mathcal{L} := \sup_{t\in\mathbb{R}_0^+}\{\mathcal{L}(t)\},$$

where

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

In this exercise, we analyze the local skew of the Refined Max Algorithm.

a) Let $P=(v_0,\dots,v_D)$ be a path of length D. Construct an execution in which (i) $H_v(0)=0$ for all $v\in V$, (ii) $H_{v_{i+1}}(t_0)-H_{v_i}(t_0)=u$ for all $i\in[D]$, (iii) $L_v(t)=H_v(t)$ for all $v\in V$ and times t, and (iv) each hardware clock runs at rate 1 after time t_0. (Hint: Simply set all message delays to d and ramp hardware clock speeds. Then show that $L_v(t)=H_v(t)$ for all $v\in V$ and t, because no node v receives a message with $L>H_v(t)-d+u$ at any time t.) (20 points)

b) Take the above execution and modify it such that a skew of $u(D-1)$ occurs at some time between nodes v_0 and v_1. (Hint: Only change message delays after time t_0, nothing else. "Pull" the nodes to the current maximum clock value one by one, starting with v_{D-1}.) (20 points)

Task 3*: Global and Local Happiness

a) Find out what the term synchronizers refers to in distributed computing!

b) Use basic techniques for synchronizers to devise an algorithm that satisfies constant amortized progress, has asymptotically optimal global skew, and has local skew $\mathcal{O}(uD)$. You may assume that $(\vartheta - 1)d \leq u$, so you don't have to worry about removing $(\vartheta - 1)d$ terms.

c) Synchronize (i.e., communicate your findings) with the other students in the TA session!