Introduction to Distributed Systems
WT 20/21

Assignment 3

Submission Deadline: Monday, 21.12.2020, 10:00

- *Submit the solution in PDF via Ilias (only one solution per group).*
- *Respect the submission guidelines (see Ilias).*

---

1 Physical Clocks [16 points]

There exist different methods to keep clocks synchronized within a distributed system. One of them is the Cristian's Clock Synchronization Algorithm.

Now, assume we have two time servers, a master M and a slave S (cf. Figure 1). The master M possesses a perfect Clock $C(t)$, whereas the slave S possesses a drifting clock $C_S(t)$. In this exercise, your task is to synchronize the slave's clock with the master's clock in different scenarios.

a) [8 points] In Figure 1 and 2, we assume that the master and the slave have the same clock frequency. Slave S starts the time synchronization at local time t_1 and receives the master's time stamps t_2 and t_3 at local time t_4. For time line 1 and time line 2, calculate the offset O and the delay d. Afterwards choose Δ so that the duration of the time adjustment is the shortest possible. Additionally calculate $C_S(t_4 + \Delta)$ as well as the clock frequency $\frac{dC_S}{dt}$ for the time interval $(C_S(t_4), C_S(t_4 + \Delta)]$. Make sure that the slave's clock is strictly monotonously increasing.

Note: Δ is an integer in clock ticks of the perfect clock.

Finally, complete the figures by drawing the ticks and corresponding time values in the time line.

Use the gray lines in between the ticks as a grid for your orientation. They are NO intermediate sub-ticks!

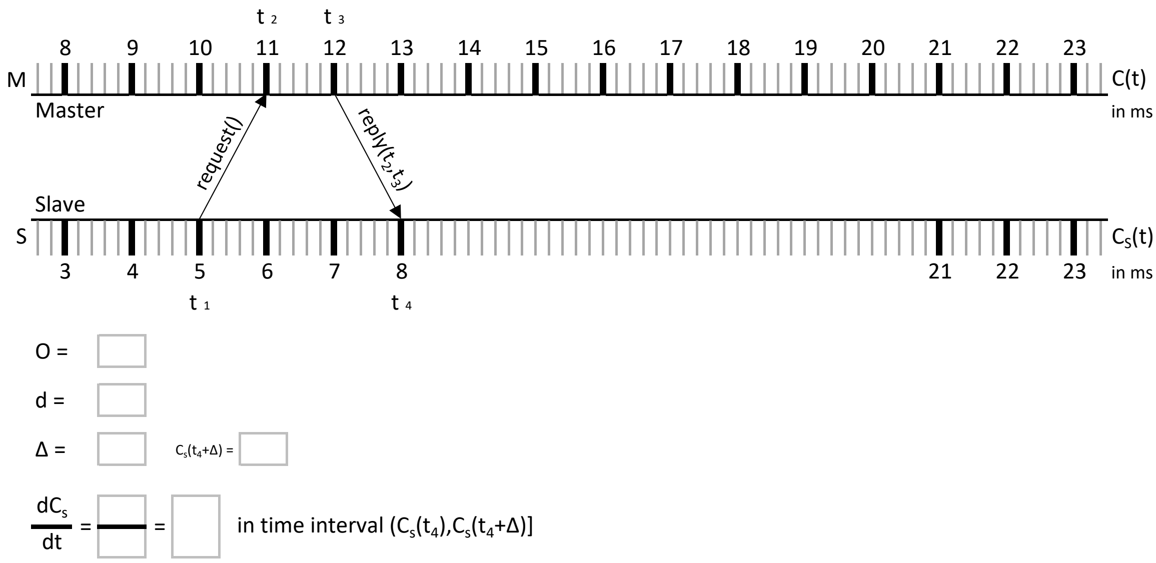

Figure 1: The time line 1 of the clock synchronization of S.

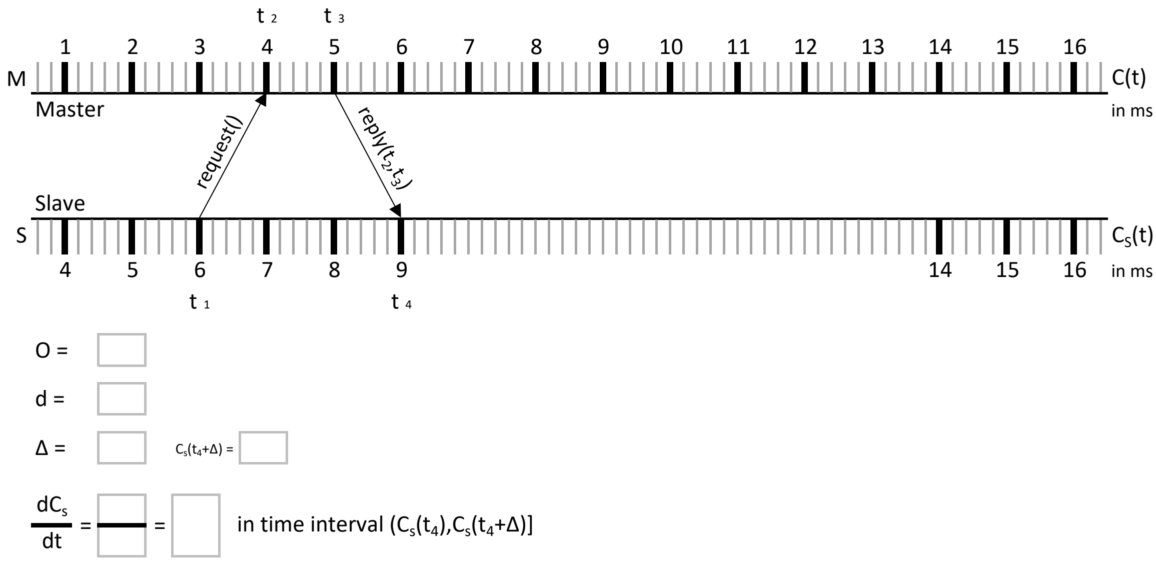

Figure 2: The time line 2 of the clock synchronization of S.

b) [8 points] In Figure 3, we assume that the master and slave have different clock frequencies.

i. Design a protocol for synchronizing the slave's clock frequency with the master's clock frequency. Assume, that the delay is constant. Show how the measured time values can be used to compute the frequency synchronization factor f, where $frequencyMaster = f \times frequencySlavebeforeSynchronization$.

ii. Now, design a protocol that performs both, phase and frequency synchronization, of the slave clock to the master clock depicted in Figure 3 using the measured time values (no additional messages, and the most recent offset). Calculate the offset O and the delay d, then choose Δ so that the duration of the time adjustment is the shortest possible and calculate $C_S(t_8 + \Delta)$ and the clock frequnecy $\frac{dC_S}{dt}$ in time interval $(C_S(t_8), C_S(t_8 + \Delta)]$. Also provide the frequency synchronization factors f_i for all your frequency adjusting steps.

Note: Δ is an integer in clock ticks of the perfect clock. Additionally, it is sufficient to synchronize phases with the precision of 1 clock tick.

iii. Finally, complete the figure by drawing the ticks and corresponding time values into the time line.

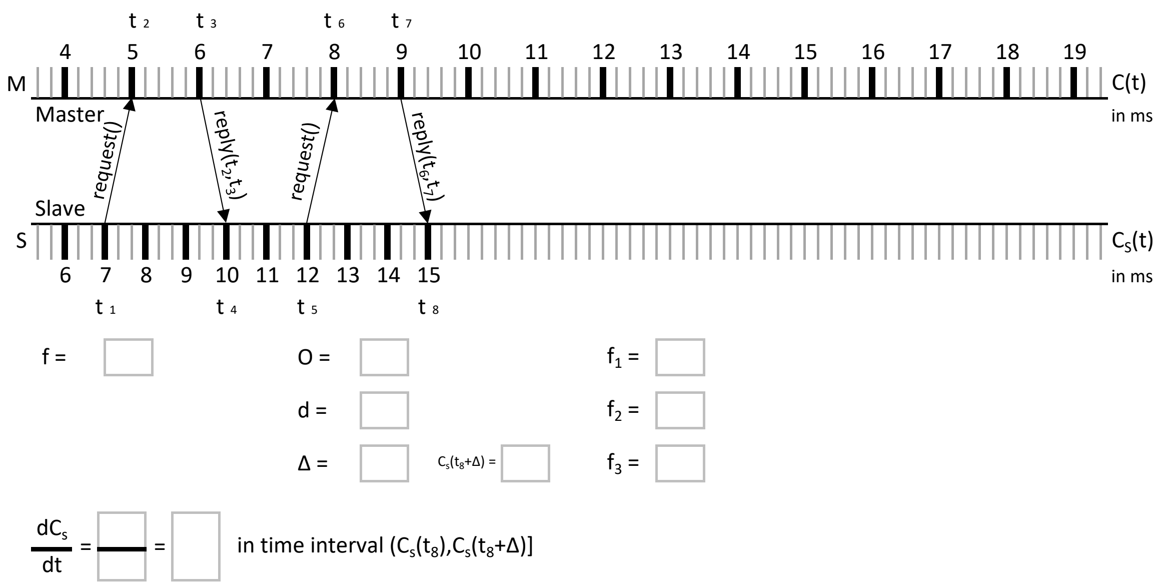

Figure 3: The time line 3 of the clock synchronization of S.

2 Logical Clocks [10 points]

a) Consider the following space-time diagram (Figure 4) that shows the local events as well as the send and receive events of the processes P_1, P_2, and P_3.

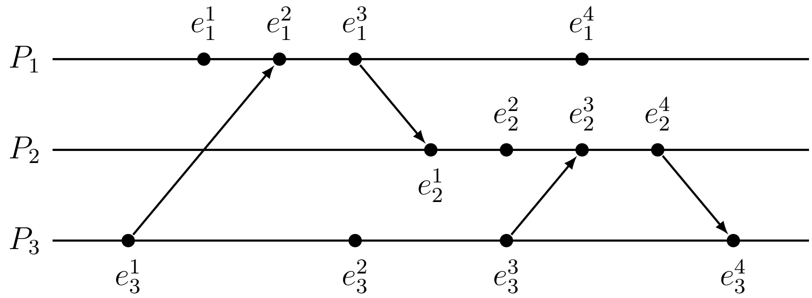

Figure 4: Distributed System with Processes $P_1 \ldots P_3$

i. [2 points] Assume that these processes use logical *scalar* clocks. Determine the timestamps of all events in the system. Each process' clock is initialized to 0.

ii. [3 points] Now, assume that the processes use logical *vector* clocks. Determine the vector timestamps of all events. Each process' clock is initialized to $(0, 0, 0)$.

iii. [1 point] Which events are not causally related to e_3^4 (i.e. $a \to e_3^4$)? Justify.

b) Consider a distributed system consisting of two processes P_1 ,P_2,P_3 using vector clocks. The events occurring in this system are given below with vector timestamps: $e_6 \begin{pmatrix}1\\0\\0\end{pmatrix}$, $e_7 \begin{pmatrix}0\\0\\2\end{pmatrix}$, $e_2 \begin{pmatrix}1\\0\\4\end{pmatrix}$, $e_4 \begin{pmatrix}4\\0\\0\end{pmatrix}$, $e_3 \begin{pmatrix}4\\2\\0\end{pmatrix}$, $e_{11} \begin{pmatrix}0\\0\\1\end{pmatrix}$, $e_8 \begin{pmatrix}5\\1\\0\end{pmatrix}$, $e_{10} \begin{pmatrix}6\\1\\0\end{pmatrix}$, $e_{12} \begin{pmatrix}0\\1\\0\end{pmatrix}$, $e_5 \begin{pmatrix}2\\0\\0\end{pmatrix}$, $e_{13} \begin{pmatrix}6\\3\\0\end{pmatrix}$, $e_1 \begin{pmatrix}1\\0\\3\end{pmatrix}$, and $e_9 \begin{pmatrix}3\\0\\0\end{pmatrix}$

Note: The order of events given in the list above is completely random.
Each process' clock is initialized to $(0, 0, 0)$.

i. [2 points] For each event $e_1 - e_{13}$, determine in which process (P_1, P_2, or P_3) it happened.

ii. [2 points] Determine for each event $e_1 - e_{13}$ whether it is a *local* event, a *send* event, or a *receive* event.

3 Global State [10 points]

Figure 5 shows a distributed system consisting of two processes P_1 and P_2.

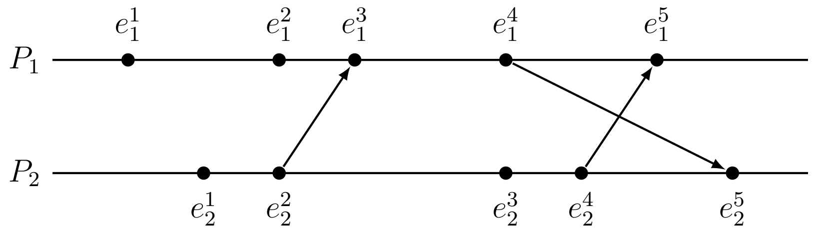

Figure 5: Distributed System with Processes P_1 and P_2

a) [4 points] Name 9 distinct pairs of concurrent events with respect to the distributed system given in Figure 5.

b) [1 point] Are the following sequences a valid linearization of the given events? Justify your answer!

  i. $\langle e_2^1, e_1^1, e_1^2, e_2^2, e_1^3, e_1^4, e_2^3, e_2^4, e_1^5, e_2^5 \rangle$

  ii. $\langle e_1^1, e_2^1, e_1^2, e_2^2, e_2^3, e_1^3, e_1^4, e_1^5, e_2^4, e_5^5 \rangle$

c) [5 points] Construct the lattice of global states (cf. Chapter 6, slides 15–16) for the system shown above. In the lattice, S_{ij} denotes the global state after i events on process P_1 and j events on process P_2. Label each edge with the event which corresponds to the state transition that edge represents.

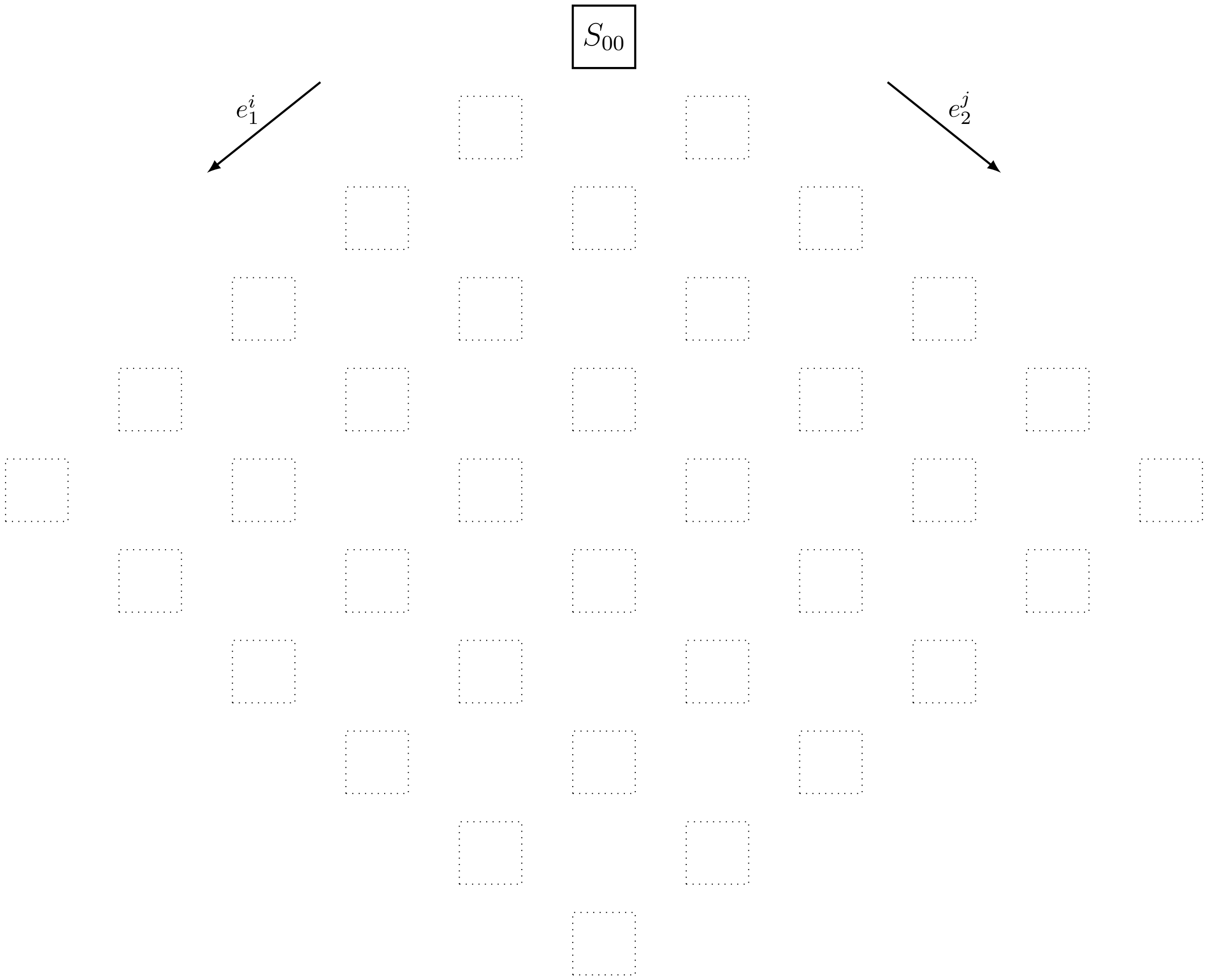

Figure 6: Lattice for Global States

4 Snapshot Algorithm [6 points]

Consider a distributed system consisting of three processes P_1, P_2, and P_3, connected by unidirectional channels $c_1, \ldots, c_4$ as given in Figure 7 and the global history of events for this system as given in Figure 8.

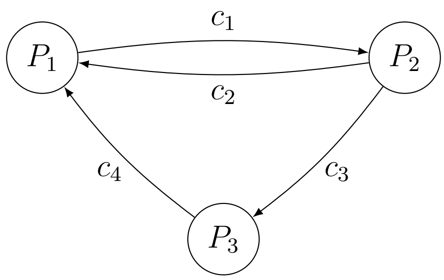

Figure 7: System of three processes executing the snapshot algorithm.

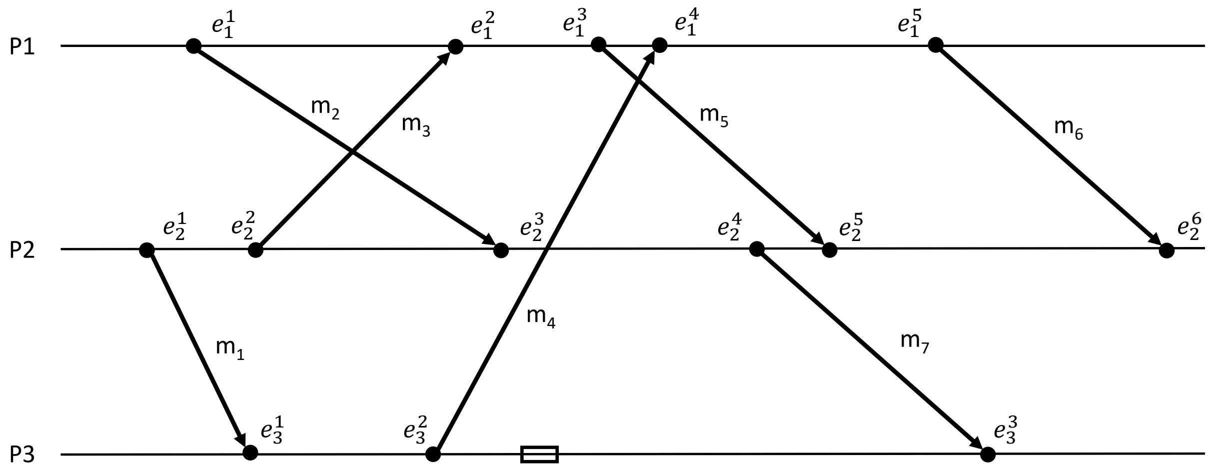

Figure 8: History for Processes $P_1 \ldots P_3$

a) [4 points] P_3 initiates the snapshot algorithm. Complete Figure 8 with a possible execution of the snapshot algorithm. Add all marker messages sent by any process. Denote where process state and channel state are saved.

Note: Make clear for which process or channel the state is being saved.

b) [2 points] For your solution of part a), give the set of messages recorded for each channel $c_1, \ldots, c_4$.