Clock Synchronization in Distributed Systems

Kasım Sinan Yıldırım * and Aylin Kantarcı
Computer Engineering Department, Ege University

Abstract- In this paper, we present the well-known shifting tool for proving the lower bounds of the distributed clock synchronization algorithms. With using this mathematical tool, we prove the lower bound on the clock synchronization error between two processors in a distributed system.

Index Terms- Distributed Clock Synchronization, Lower Bounds, Shifting.

I. Introduction

Distributed systems consist of a collection of distinct processes (called *nodes*) which are spatially separated and which communicate with one another by exchanging messages [1]. The clocks of the nodes tick at different rates and as a result these clocks can drift apart. Thus, the clocks in the distributed system may not remain always synchronized although they might be synchronized when they start.

The lack of global notion of time poses serious problems for the correct operation of the distributed applications and protocols which need synchronized clocks. Consequently, it is mandatory to provide a distributed clock synchronization algorithm whose objective is to ensure that the nodes are able to acquire a common notion of time. Clock synchronization algorithms are based on exchanging clock information among the nodes and try to eliminate the effects of non-determinism in the message delay and data processing time.

In this paper, we are interested in the lower bound on the clock synchronization error between two processors in a distributed system [3][4]. The system model which we use in the rest of this paper is presented in Section II. We prove the best achievable clock synchronization error in a distributed system in Section III. Finally, Section IV is the conclusion.

II. The System Model

In this section, we describe the formal model which will be used for the analysis of the clock synchronization algorithms. We describe the model of communication and clocks in the system.

A. The Clock Model

We assume that each node in the distributed system is equipped with a *physical clock* which is used for measuring the time. We denote the reading of physical clock at real time t as $H(t)$. The *rate* $h(t)$ of the physical clock is the first derivate of $H(t)$. The *clock drift* $\rho(t)$ occurs where the rate of the physical clock is different from the standard rate 1.

$$h(t) = \frac{dH(t)}{dt} \tag{1}$$

$$\rho(t) = h(t) - 1 \tag{2}$$

We assume that the physical clocks of the nodes have bounded drifts:

$$1 - \rho \le \rho(t) \le 1 + \rho \tag{3}$$

A clock synchronization algorithm cannot directly modify physical clocks and alternatively it uses the physical clock and the messages it receives from the other nodes to compute a *logical clock value* $L(t)$. We assume that nodes in the system also maintain a logical clock. The logical clock is not allowed to run backwards. The clock synchronization algorithm can either increase the logical clock or leave it at the current value.

B. The Communication Model

We consider a set of *n* nodes located in the Euclidean plane that communicate with each other through wireless broadcasts by exchanging messages. We assume that there exists a path between every pair of nodes which means communication network is *connected*. We assume that each link is reliable, FIFO, symmetrical and has bounded delay. We model communication network as a graph *G={V,E,L,H}* where *V={1..n}* represents the nodes and $E \in VxV$ represents the links between the nodes. Any node *i* can communicate with any node to which it is directly connected and these nodes are referred to as the neighbors of that node, which is denoted by *N*. *L(i,j)* denotes the minimum message delay on the edge connecting nodes *i* and *j* and *H(i,j)* denotes the maximum message delay. The difference *H(i,j)- L(i,j)* is the uncertainty on that edge.

C. The System of Nodes with Clocks

Occurrences that take place in the system are called *events* and are denoted by ϕ. We assume that the events in the system are *computation events* and *delivery events* representing the delivery of message between nodes. A *history* φ of a node contains pairs $(\phi, H(t))$ for all events and their hardware clock readings at the real-time which the event occurred. An execution α is a set of *n* histories, one for each node. An execution α is *admissable* for the communication network *G* if every message between any two nodes has delay within the interval [*L(i,j),H(i,j)*]. Two executions α_1 and α_2 are *indistinguishable (equivalent)* if each node in the system has the same history in α_1 and in α_2.

D. The Clock Synchronization Problem

Given the formal system definition, we now formally state the *clock synchronization problem*. The clock synchronization problem deals with the internal synchronization of the nodes and states the precision requirement on the difference between the logical clock values of any two nodes which is called the *global skew*.

Definition 1. **Achieving ε-Synchronized Clocks**: In every execution that is admissable for *G*, there exists a real time t_e such that the algorithm has terminated and for all nodes *i* and *j* and all $t \geq t_e$, it holds that $|L_i(t) - L_j(t)| \leq \varepsilon$.

III. THE BEST ACHIEVABLE CLOCK SYNCHRONIZATION ERROR

After presenting the system model, we will now focus on the best achievable clock synchronization error in a system with only two processors *p* and *q*. We will define the *shifting lemma* which is an important mathematical tool for proving the best achievable synchronization error for the clock synchronization algorithms. With shifting, we start with one execution and modify the real-time occurrence of the events by slowing down or speeding up the clocks of the nodes and obtain another equivalent and admissable execution. The processors cannot tell any difference because events still happen at the same hardware clock times. Fig. 1. shows shifting of the processor *p* earlier and later. The shifting process is formalized with the following lemma.

Lemma 1. **Shifting** [3]. Let α be an execution with hardware clock H^{α} and let x be a real number. Then $\beta = shift(\alpha, x)$ is an execution with hardware clock H^{β}, where

(a) $H^{\beta}(t) = H^{\alpha}(t) - x$ for all *t*

(b) if the delay of a message in α from *p* to *q* is *d*, then the delay of this message in β is *d*- *x*.

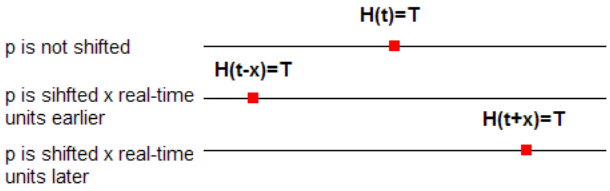

Fig. 1. Shifting of processor *p* earlier and later.

A. The Lower Bound

We will now prove the lower bound on the clock synchronization error between two clocks. For our analysis, we assume that the nodes *p* and *q* have perfect clocks (do not drift). We also assume that every message exchanged between p and q has a delay within the interval [*d-u*,*d*].

Let Φ be a distributed clock synchronization algorithm that achieves ε-synchronized clocks between the nodes *p* and *q* and terminated at time *t*. Consider the two executions shown in Fig. 2. In the first execution (α), every message from *p* to *q* has delay *d-u* and from *q* to *p* has delay *d*. We shift the processor *p* *u* real-time earlier in the second execution (β) where every message from *p* to *q* has delay *d* and from *q* to *p* has delay *d-u*. Since all the events occurring at both processors has the same hardware clock readings, these two executions are equivalent and indistinguishable from the point of view of the nodes *p* and *q*.

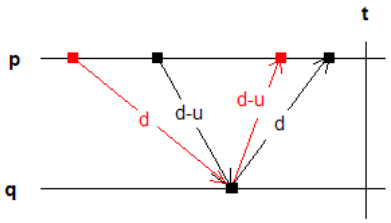

Fig. 2. Two executions with two processors *p* and *q*

Since the algorithm achieves ε-synchronized clocks, at time *t* in the first execution we have that:

$$L_q^{\alpha}(t) - L_p^{\alpha}(t) \leq \varepsilon \quad (4)$$

After shifting, the algorithm must achieve the same accuracy for the synchronized clocks. Since $L_p^{\beta}(t) = L_p^{\alpha}(t) + u$ and $L_q^{\beta}(t) = L_q^{\alpha}(t)$, we again have:

$$L_q^{\beta}(t) - L_p^{\beta}(t) = L_q^{\beta}(t) - L_p^{\beta}(t) + u \leq \varepsilon \quad (5)$$

If we combine inequalities (4) and (5), we get that $\varepsilon \geq u/2$. This result states that there is not any distributed clock synchronization algorithm that will achieve a worst-case synchronization error lower than *u/2* in a network with two nodes where the hardware clocks of the nodes do not drift.

IV. CONCLUSION

In this paper, we presented the lower bound techniques used for proving the worst-case achievable synchronization in a network with two processors. Lundelius and Lynch [4] improved this lower bound for a fully connected network with *n* nodes. They showed that it impossible to synchronize the clocks of *n* processes any more closely than *u(1-1/n)*. Later, Biaz and Welch [5] proved that for any communication network *G*, the best achievable synchronization error is half of the diameter of the communication network.

V. ACKNOWLEDGMENT

Kasım Sinan YILDIRIM acknowledges The Turkish Scientific and Technical Research Council (TUBITAK) for supporting this work through a domestic PhD scholarship program BAYG-2211).

VI. REFERENCES

[1] L. Lamport, "Time, clocks, and the ordering of events in a distributed system." *Commun. ACM* 21, 7 (Jul. 1978), 558-565.

[2] A. Kshemkalyani and M. Singhal, "*Distributed Computing: Principles, Algorithms, and Systems*.", Cambridge University Press, 2008

[3] H. Attiya and J. Welch, "*Distributed Computing: Fundamentals, Simulations and Advanced Topics*.", John Wiley & Sons, 2004

[4] J. Lundelius and N. Lynch, "An upper and lower bound for clock synchronization", Information and Control, Volume 62, Issues 2-3, August-September 1984, Pages 190-204

[5] S. Biaz and J. L. Welch, "Closed form bounds for clock synchronization under simple uncertainty assumptions." Elsevier Information Processing Letters, 80:151_157, 2001.