Virtual Time Synchronization for Multimedia Ad Hoc Networks

Anelise Munaretto\textsuperscript{1,2}, Mauro Fonseca\textsuperscript{1,4}, Khaldoun Al Agha\textsuperscript{2,3}, and Guy Pujolle\textsuperscript{1}

\begin{flushright}
\textsuperscript{1}LIP6 Laboratory, University of Paris VI, Paris, France
\textsuperscript{2}LRI Laboratory, University of Paris XI, Orsay, France
\textsuperscript{3}INRIA Laboratory, Rocquencourt, Le Chesnay, France
\textsuperscript{4}PPGIA Laboratory, PUC-PR, Curitiba, Parana, Brazil
\end{flushright}

E-mail: \{anelise.munaretto, mauro.fonseca, guy.pujolle\}@lip6.fr; alagha@lri.fr

Abstract—We propose a novel protocol to deal with the synchronization problem for multimedia ad hoc networks. Our proposal eliminates the need for hard clock synchronization by implementing a virtual scheme that relies on the desynchronization between nodes. Such a decision is made because of the dynamic nature of the network topology. Therefore, the proposed algorithm computes the clock offsets between a node and each one of its one hop neighbors. We formulate our algorithm by deriving simple expressions combining stochastic and measured parameters. The proposed algorithm is validated through implementation and the results show that high levels of synchronization are achieved.

I. INTRODUCTION

An ad hoc network is expected to operate in a network environment where some or all the nodes are mobile, which contrasts with traditional wireline or wireless networks. In this dynamic environment, network functions should run in a distributed fashion, since nodes might suddenly join or leave the network. The success of such networks depends on the deployment of multiple and diverse applications, e.g. multimedia services.

Network synchronization deals with the distribution of time and frequency across a network of clocks often spread over a wide geographical area \cite{7}. Network synchronization plays a main role since it is used to determine the quality of most services offered by the network. However, the importance of network synchronization is often undervalued and likewise how to resolve QoS degradation caused by a probably desynchronization.

Supporting multimedia services implies the consideration of various QoS requirements, often given in terms of bandwidth or delay. In order to measure delays, a synchronized network would be the ideal scenario, which would avoid flooding and control traffic overhead required in unsynchronized networks. Nevertheless, ad hoc networks are not expected to supply any centralized reference node. Thereby, classical clock synchronization algorithms like NTP \cite{1} are not appropriate. Moreover, hardware clocks are imperfect due to the clock drift variation, which results in a monotonically increasing desynchronization. The clock drift can be defined as the accuracy variation or random oscillation about the same clock.

For instance, the authors in \cite{2} show that the 802.11 standard protocol may have clock drifts over 5000 \( \mu \)s for 300 stations while in the ad hoc mode. In this way, there has been much work on the clock synchronization problem.

A novel protocol introduced in \cite{3} presents a synchronization algorithm for a certain class of applications of sparse ad hoc networks using timestamp computation. The proposed scheme piggybacks round trip time measurements into ordinary data packets sent by other processes. This achieves time synchronization without any significative overhead in the network. Nevertheless, the synchronization is local and rather short-lived.

In \cite{5} researches present TPSN (Timing-sync Protocol for Sensor Networks) that aims at providing network-wide time synchronization. Nevertheless, a hierarchical topology must be created where every node is assigned to a level. They use the “always-on” model where every node maintains a clock that is synchronized with respect to a reference node in the network.

In wireless links, bandwidth and latency are often different in the uplink and downlink directions. However, the authors in \cite{4} have shown that in the specific case of WLANs, links are symmetric. We propose a new approach to treat the synchronization problem for multimedia ad hoc networks.

A. Background

Time (clock) synchronization problem has been deeply investigated in Internet and LANs. Several technologies, such as GPS, have been used to supply overall synchronization in networks. Complex protocols, like NTP, have been developed that have kept Internet’s clocks synchronized.

Accurate time synchronization among distributed hosts is essential to realizing both the well managed, QoS-aware Internet and advanced distributed applications on it. Accurately synchronized clocks on distributed hosts can play an important role in network management and research, specially in one-way packet delay measurement \cite{6} that gives us helpful information on network internal states as well as end-to-end network performance. Additionally, synchronized clocks offer not only precise time or time-interval information but also on the order in which distributed events occur, and hence new
distributed applications will involve clock synchronization in the order under milliseconds on each end-user.

Network Time Protocol (NTP) is wide deployed in the Internet to synchronize dispersed clocks to every additional or to the instance server having a precise clock. Since it estimates the clock offset and inclination based on one-way delay measurement across the network, the accuracy of synchronization depends on the network (the paths) between two hosts, while it can maintain time within millisecond-order in various networks. The clock offset can be defined as the time clock divergence among nodes.

NTP has been successfully able to synchronize the clocks in Internet to an accuracy of the order of the milliseconds but the time synchronization requirements for ad hoc networks can be much more stringent than that, ranging into the order of a few microseconds. Our framework can be viewed as a practical, more accurate and a flexible than NTP.

Special devices for clock synchronization, such as the Global Positioning System (GPS), are also available and can synchronize a local clock to the standard time very exactly. Nevertheless, it is not ubiquitous because it requires a particular environment like an antenna. Mainly when “everything on IP”, since as “everything” has its clock, distributed clock synchronization based on network protocols is still significant. One of main factors increasing errors in estimate the clock offset between two hosts is the asymmetric paths problem. Since existing methods (like NTP) do not take the asymmetric paths into account, the larger difference between the one-way delay of the forward path and that of the backward path causes the supplementary inaccuracy in the clock offset estimation.

II. A VIRTUAL TIME FRAMEWORK

We present a time synchronization protocol for ad hoc networks that works on the conventional approach of sender-receiver synchronization. We argue that for ad hoc networks, the classical approach of doing a handshake between a pair of nodes is a better approach of synchronizing a set of receivers. In ad hoc networks, clock synchronization might not be needed all the times.

Our proposal consists of two major phases. In the first phase, a dialogue is established between the arriving node and its neighbors. Such a message exchange could be an integrating part of a proactive routing protocol, where nodes send control messages periodically. In this case, the main advantage is that there is no need for additional synchronization messages. The frequency of these exchanges is varying in time. The second phase is based on the measured parameters during the first phase. Let \( \Delta \) be the clock offset between a node and one of its neighbors. The algorithm stores and computes an average \( \Delta \) for each neighbor. The main idea is to find and store the perceived offsets in a clock offset table. This table yields virtual time synchronization between nodes.

A. Formulation

Fig. 1 depicts the notation used in our proposed algorithm.

Fig. 1. Scenario.

Independently of time synchronization any node can compute \( RTT = T_2 - T_0 \). Consider \( V_{T_1} = T_1 + \Delta \), as the timestamp at node B when the request message comes in. Note that \( V_{T_1} \) is sent in the reply message and hence known by the node.

Let be \( D_{snd} \) the delay to send a packet from node A to node B. The same, \( D_{rcv} \) represents the delay to receive a packet from node A to node B. Note that the bar notation used in our formulas represent a measured value (real value) and not an average value.

\[
RTT = D_{snd} + D_{rcv} \tag{1}
\]

\[
D_{snd} = T_1 - T_0 \tag{2}
\]

\[
D_{rcv} = T_2 - T_1 \tag{3}
\]

\[
V_{T_1} = T_1 + \Delta \tag{4}
\]

Let \( \theta \) be the latency between nodes A and B.

Assumption: We assume that nodes are similar (use the same link access protocol) and that the long-term medium conditions follow the same Probability Density Function (p.d.f) for both nodes A and B, thus

\[
P_A(D_{snd} = \theta) = P_B(D_{snd} = \theta) = ... \tag{5}
\]

Based on the assumption, all nodes have the same probability to have a sending delay equal to \( \theta \), then:

\[
P_A(D_{rcv} = \theta) = P_B(D_{rcv} = \theta) = ... \tag{6}
\]
Similarly, all nodes have the same probability to have a receiving delay = $\theta$. According to Fig. 1, we have that $D_{\text{snd}}A = D_{\text{rcv}}B$ and that $P(D_{\text{snd}}A) = P(D_{\text{rcv}}A)$. Hence, the average delay $\mu_{D_{\text{snd}}}$ during $n$ units of time, where $\lim_{n \to \infty}$ is represented by

$$\mu_{D_{\text{snd}}} = \sum_{i=0}^{n} D_{\text{snd}}i \cdot P(D_{\text{snd}}i)$$

(7)

Conforming to our assumption, at the long-term all delays have the same probability to happen, so

$$\mu_{D_{\text{snd}}} = \frac{1}{n} \sum_{i=0}^{n} D_{\text{snd}}i$$

(8)

The $\text{RTT}$ is thus represented as:

$$\frac{1}{n} \sum_{i=0}^{n} \text{RTT}_i = \frac{1}{n} \sum_{i=0}^{n} D_{\text{snd}}i + \frac{1}{n} \sum_{i=0}^{n} D_{\text{rcv}}i$$

(9)

or

$$\frac{1}{n} \sum_{i=0}^{n} \text{RTT}_i = \frac{1}{n} \sum_{i=0}^{n} D_{\text{rcv}}i + \frac{2}{n} \sum_{i=0}^{n} D_{\text{rcv}}i$$

(10)

$$\frac{1}{n} \sum_{i=0}^{n} \text{RTT}_i = \frac{2}{n} \sum_{i=0}^{n} D_{\text{rcv}}i$$

(11)

$$\sum_{i=0}^{n} \frac{\text{RTT}_i}{2} = \sum_{i=0}^{n} D_{\text{rcv}}i$$

(12)

Applying in 3 we have

$$\sum_{i=0}^{n} \frac{\text{RTT}_i}{2} = \sum_{i=0}^{n} T_{2i} - T_{1i}$$

(13)

And from 4 we have

$$\sum_{i=0}^{n} \frac{\text{RTT}_i}{2} = \sum_{i=0}^{n} T_{2i} - (V_{T1i} - \Delta)$$

(14)

$$\sum_{i=0}^{n} \frac{\text{RTT}_i}{2} = \sum_{i=0}^{n} T_{2i} - V_{T1i} + \Delta$$

(15)

$$\frac{1}{n} \sum_{i=0}^{n} \Delta = \frac{1}{n} \sum_{i=0}^{n} \left( \frac{\text{RTT}}{2} - T_{2i} + V_{T1i} \right)$$

(16)

Finally:

$$\Delta = \frac{1}{n} \sum_{i=0}^{n} \left( \frac{\text{RTT}}{2} - T_{2i} + V_{T1i} \right)$$

(17)

We verify that $\Delta$ is not only a function of $\frac{\text{RTT}}{2}$ but also of $T_2$ and $T_1$. This is not a stationary process, because of the deviation of the clock variation and the clock drift over the time. Thereby, we propose that in each period of time, a synchronization may be done without the need of the sending messages in order to calculate the $\Delta$.

We use the same model as proposed on Reference-Broadcast Synchronization (RBS) [8]. In this model, though every node maintains an individual clock, these clock are not synchronized with respect to each other. Instead, every node stores information about the relative drift between its clock and the clock of any other node in the networks. In our case, only nodes belonging to one hop neighbors maintain a relative clock.

### III. Performance Measurements

The frequency of the messages exchange represents the number of the sent messages per second by a node. In order to measure a stabilization time and estimate an ideal frequency of these exchanges, we implement our protocol for two nodes. The scenario is composed by a Pentium III 850 MHz and an Athlon XP 1500+ connected by 802.11b cards.

Fig. 2 depicts the results varying from 1 to 20 messages exchanged per second. As expected, we observe that the best result is achieved using 20 messages per second. However, such a frequency intensifies the overhead in the network. Nevertheless, 1 message per second is synchronized later. Therefore, we conclude that 5 messages per second provides a proper compromise between the overhead in the network and the time to synchronization.

The second measurement result, illustrated in Fig. 3 the behavior of the time synchronization varying the time interval to send 5 messages, i.e. 51 means sending of 5 messages in 1 second, 52 means sending of 5 messages in 2 seconds, and so on. These results vary in a negative scale, representing the clock difference between the referenced nodes. We can clearly verify that as higher as the frequency to send 5 messages is, lower is the synchronization time.

This figure shows that after 400 s the algorithm stabilizes in a constant value. However, this constant includes not only the clock offset $\Delta$. In fact, since all hardware clocks deal with clock drift, the resulting value less than 45 $\mu$s includes the clock offset $\Delta$, the clock drift specific to our scenario, and the processing time at the nodes during the messages exchange. Moreover, after the stabilization time, we can mitigate the
messages exchange and periodically update the clock offset table with the stabilized value. Furthermore, our ongoing performance evaluation aims to assess the improvement achieved by integrating our proposed algorithm in routing protocol.

The stabilization period presented in measurements results represents the duration of time necessary in order to achieve a clock synchronization. It is easier computed in our implementation because we know exactly the clock offset.

IV. CONCLUSION

We presented a distributed virtual time synchronization protocol for multimedia ad hoc networks. Our proposal eliminates the need for hard clock synchronization by implementing a virtual scheme that relies on the desynchronization between nodes. The main idea is to find and store the perceived offsets in a clock offset table. This table yields virtual time synchronization between nodes. We formulated the proposed algorithm combining stochastic and measured parameters. Our decentralized scheme achieves high level of synchronization under a low communication cost.

REFERENCES