Evaluation of Kalman Filtering for Network Time Keeping

Aggelos Bletsas, *Associate Member, IEEE*

***Abstract*—Time information is critical for a variety of applications in distributed environments that facilitate pervasive computing and communication. This work describes and evaluates a novel Kalman filtering algorithm for end-to-end time synchronization between a client computer and a server of "true" time [e.g., a Global Positioning System (GPS) source] using messages transmitted over packet-switched networks, such as the internet. The messages exchanged have the network time protocol (NTP) format, and the algorithm evaluated, is performed only at the client side. The Kalman filtering algorithm is compared to two other techniques widely used, based on linear programming and statistical averaging, and the experiments involve independent consecutive measurements (Gaussian case) or measurements exhibiting long-range dependence (self-similar case). Performance is evaluated according to the estimation error of frequency offset and time offset between client and server clock, the standard deviation of the estimates and the number of packets used for a specific estimation. The algorithms could exploit existing NTP infrastructure, and a specific example is presented.**

I. Introduction

In this paper we evaluate a novel Kalman filtering algorithm for end-to-end time synchronization between two computers exchanging messages over a packet-switched network such as the internet. The Kalman filtering algorithm is compared to two other techniques widely used, based on linear programming and statistical averaging. We are particularly interested in the calculation of frequency offset and time offset between the clock of a client computer and the clock of a remote server. The server acts as a source of true time.

Accurately synchronized clocks enable services and provide the basis for efficient communications. Autonomous sensor array operation is facilitated by accurate time stamps [1]. Global positioning system, as well as proposed ultra-wide band urban and intra-building location systems [2] rely on precise timing measurements. Internet performance can be evaluated from accurate measurement of the delay between various nodes in the network. Various important internet protocols such as TCP could benefit from accurate time keeping [3].

In this work, we exploit the network time protocol (NTP) [4] messages between a client and a single server, in three different algorithms and evaluate their performance. However, the algorithms do not depend on the details of the NTP message format, and other formats could be adopted with minor modifications. The NTP is a hierarchical client-server synchronization scheme widely used and can provide for accuracies on the order of milliseconds under the assumption that there is a reasonably short round trip time between client and server. The NTP also incorporates connections to multiple servers for increased reliability. The following discussion is about the algorithms for the local clock parameters estimation when an internet connection is used to a single time reference server. These parameters then could be used to steer the local clock according to hardware and operating system details.

The rest of the paper is organized as follows. In Section II we introduce the terminology used throughout this work and proceed to formulate the problem. We present prior art and justify the selection of three algorithms compared in this work. In Section III we analytically present algorithms based on Kalman filtering, linear programming, and averaging of time differences. In Section IV we investigate their performance for the Gaussian case (no dependence between successive measurements) and for the self-similar case (long-range dependence). We conclude in Section V.

II. Background

A. Clock Basics

Using the representation $C(t)$ for a clock reading and $T(t) = t$ for true time, the following definitions are presented:

- **Time offset:** the difference between the time reported from a clock and the true time: $C(t) - T(t) = C(t) - t$. In this paper we will refer to the time offset calculated for $t = 0$ as θ and for $t \neq 0$ as x.
- **Frequency offset** (also referred as skew): the difference in frequencies between a clock and the true time: $C'(t) - T'(t) = C'(t) - 1$. In this work, we will refer to frequency offset as $\phi - 1$.
- **Drift:** the long-term frequency change of a clock. Drift is caused by changes in the components of the oscillator and its environment.

Typical quartz oscillators (without any type of temperature compensation) exhibit frequency offsets on the order of a few parts per million. For example a 10 ppm oscillator will introduce an uncertainty (i.e., error) of 36 ms in 1 hour. Cesium beam atomic clocks, however, exploiting

Manuscript received March 8, 2004; accepted February 18, 2005.

The author is with MIT Media Laboratory, Cambridge, MA 02139 (e-mail: aggelos@media.mit.edu).

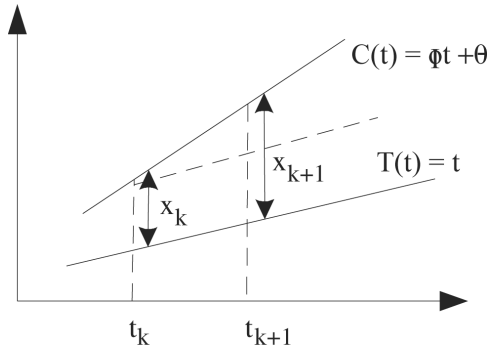

Fig. 1. Frequency offset $\phi - 1$ and time offset θ of $C(t)$, compared with the source of true time $T(t)$.

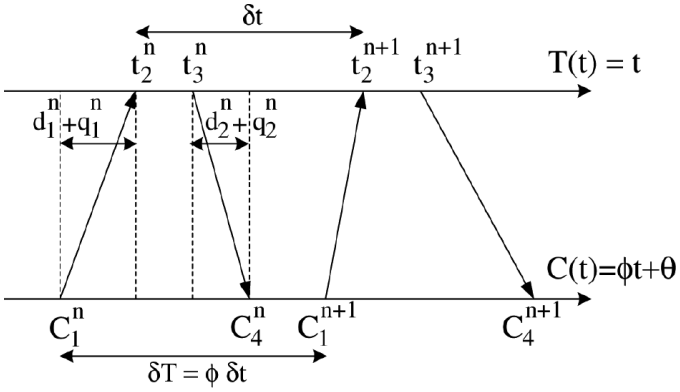

Fig. 2. Exchanging timestamps between client and time server. Notice that a time difference of δt according to server clock is translated to $\phi\delta t$ according to client clock.

the stabilities of the quantum world perform better with uncertainties close or smaller than 1 ns in 24 hours.

Modeling a clock as a piecewise linear function of time is a reasonable step as any function can be approximated in a similar manner. The client should estimate only two parameters, namely, the time and frequency offset θ, $\phi-1$, respectively, compared to the source of true time $T(t)$ as depicted in Fig. 1, as only two parameters are needed to define a line.

However, for this model to be realistic, it is important to keep the duration of the measurement process as small as possible, before ϕ and θ at the client clock are modified. The parameters ϕ, θ change with a rate related to the clock drift; and, it has been found that for most free running oscillators used in current computer systems, this change happens at intervals on the order of 1–2 hours or more [5]. That is reasonable to expect because macroscopic factors that heavily influence crystal oscillators, such as temperature, change no faster than that rate.

A statistical tool that provides a stationary measure of the stochastic behavior regarding time deviation residuals and their associated frequency fluctuation estimates is the Allan variance [6]. Allan variance associates frequency fluctuation estimates with specific observation duration and, therefore, could be used to quantify how often the above clock parameters change. For an excellent review of oscillators, Allan variance, time and frequency metrology, the interested reader could refer to [7].

B. Problem Formulation

After describing the clock nomenclature followed in this work, we are ready to formulate the problem. The client clock $C(t)$ is synchronized to a time source $T(t) = t$ when both frequency offset ϕ and time offset θ are estimated.

The client timestamps ($C(t_1)$) a User Datagram Protocol (UDP) packet according to each own clock $C(t)$ and sends the message to a time source server that timestamps the packet upon reception and retransmission (t_2, t_3, respectively) back to the originating client (Fig. 2). The client timestamps the message again upon reception and, therefore, acquires a set of four timestamps: $(C(t_1), t_2, t_3, C(t_4))$. For convenience, we will notate $C(t_1)$ as C_1 and $C(t_2)$ as C_2 from now on. The same process can be repeated for a set of N consecutive messages. Therefore, we should answer the following questions:

- What is the optimal processing of N messages $(C_1^1, t_2^1, t_3^1, C_4^1)$, $(C_1^2, t_2^2, t_3^2, C_4^2), \ldots,$ $(C_1^N, t_2^N, t_3^N, C_4^N)$ so as to obtain unbiased estimates with minimum error?
- What is the cost of obtaining estimates of ϕ and θ in terms of bandwidth spent (number N, inter-departure time between packets)?
- Do the algorithms used in the estimation of ϕ, θ impose special restrictions in the operation of client (or server) operating system (i.e., are there any major nonalgorithmic modifications in the operation of existing client/time server daemons)?

The number of packets N exchanged between client and server (Fig. 2) is a crucial parameter of any algorithm eventually adopted, considering the heavy load of current internet time servers—on the order of 1000–1200 requests per second and increasing every year [8]. Moreover, the inter-departure intervals of the NTP-like messages should not be very large because closely spaced packets ensure that the clock parameters are not changing during the measurements from N packets.

Also, we need to emphasize that the queuing delay q_1 across the forward path (from client to server) is never constant and generally different from the queuing delay q_2 across the reverse path (from server to client) (Fig. 2). Moreover, because the messages are carried through UDP packets, the forward and reverse routes could be physically different; therefore, the propagations delays[1] d_1, d_2 could be unequal across the forward and reverse paths:

$$d_1 + q_1 \neq d_2 + q_2. \tag{1}$$

[1]Time needed for the first bit to arrive at the destination as opposed to transmission delay that is related to the speed of the link.

C. Prior Art on Client-Server Schemes

The NTP estimates the time offset using the four timestamps of a message, according to the following equation:

$$\hat{x}_n = \frac{C_1^n - t_2^n - t_3^n + C_4^n}{2}. \tag{2}$$

Because the round-trip time (rtt) is on the order of a few ms, the contribution of the frequency offset on the error for a single measurement is negligible (e.g., a 10 ppm oscillator for a 10 ms rtt exhibits 0.1 μs that is on the order of "noise" due to the operating system) and, therefore, excluded from (2). The frequency offset can be estimated using several measurements of x as depicted in Fig. 1.

From a closer look at (2), NTP estimates are erroneous by a quantity proportional to half the difference between forward and reverse path delays (asymmetry).

$$\hat{x}_n = x_n + \frac{d_2^n + q_2^n - d_1^n - q_1^n}{2} \Rightarrow \tag{3}$$

$$\hat{x}_n = x_n + w_n. \tag{4}$$

That is why the NTP error is upper bounded by half the round-trip time. If we make the assumption that the asymmetry, depicted as "noise" w_n in (4) for the n-th NTP message, is an additive white Gaussian, zero-mean random variable, then the estimate of (2) is the maximum likelihood estimate, equivalent to the efficient[2] minimum variance, unbiased estimator for this particular case, according to the Gauss-Markov theorem. However, the asymmetry is not always Gaussian, as we will discuss in the following sections.

Line-fitting techniques, based on the median slope calculated from averaged one-way delay measurements [9] or linear programming [10] are alternative proposals for frequency offset estimation. The linear programming technique proposed in [10] is revisited with a slightly different derivation that provides not only for frequency offset $(\phi - 1)$ estimation but also for time offset estimation (θ).

In the Gaussian case, averaging N measurements from (2) can improve the estimates (decreasing the standard deviation of the estimate) by a factor of $\sqrt{N}$. This is an idea exploited in the client-server synchronization schemes deployed by the National Institute of Standards and Technology (NIST) using dedicated phone lines [11] or the internet [5]. A variant of this method is discussed in this work. A similar approach based on averaging is also investigated in [12].

Kalman filtering is an attractive alternative for clock parameter estimation [13] because Kalman filters are the optimal linear estimators for the Gaussian case, i.e., the linear estimators that minimize the mean square error (MSE) [14]. As we will see in the next section, the problem can be formalized using the Kalman filtering notation and due to the optimality property (at least for the Gaussian case) excels over a range of recursive estimators like phased lock loops [13].

The optimality and the appealing recursive nature of Kalman filtering, the intuitive structure (as explained below) of the linear programming technique and the simplicity of the averaging technique (referred as averaged time differences (ATD)) as well as its wide deployment, were the reasons behind the selection of the above algorithms for comparative performance evaluation.

[2]The efficient estimator, when it exists, achieves the minimum variance of the estimate, equal to the Cramer-Rao bound.

III. The Algorithms

A. Kalman Filtering

The motivation behind the adoption of Kalman filtering stems from a simple observation: a time interval δt according to true time is translated to $\phi\,\delta t$ according to client clock (Fig. 2). Therefore, it is sufficient for the client to send messages at constant intervals δT measured according to its local clock and estimate the inter-arrival intervals at the server, using the timestamps $\{t_2^n\}$ that correspond to true time. Variation of forward and reverse one-way delays are interpreted as noise in the estimation process.

With the above, the formulation of the problem using Kalman filtering becomes clear: the client sends the NTP packets at constant intervals δT and estimates the inter-arrival interval $s = \frac{\delta T}{\phi}$ in the presence of network delay variations v, exploiting the measured inter-arrival intervals $y_n = t_2^{n+1} - t_2^n$ for $n \in [1..N]$.

The measurement and state model of the Kalman filter easily follow (Fig. 2):

$$y_n = t_2^{n+1} - t_2^n \tag{5}$$

$$= t_1^{n+1} + d_1^{n+1} + q_1^{n+1} - (t_1^n + d_1^n + q_1^n) \tag{6}$$

$$= \underbrace{t_1^{n+1} - t_1^n}_{\delta t} + \underbrace{(d_1^{n+1} + q_1^{n+1})}_{e^{n+1}} - \underbrace{(d_1^n + q_1^n)}_{e^n} \Rightarrow$$

$$y_n = \delta t + e^{n+1} - e^n = \delta t + v_n, \tag{7}$$

$$s_n \equiv \delta t = \frac{\delta T}{\phi},\ n \in [1..N] \Rightarrow \tag{8}$$

$$y_n = s_n + v_n,\ \text{measurement model}, \tag{9}$$

$$s_{n+1} = s_n + w_n,\ \text{state model}. \tag{10}$$

The measurement noise v_n accounts for the variation of travel time, when the NTP message is transmitted from client to server; and it is assumed a zero mean process throughout this work. This is the type of noise that depends on the network path between client and server. Its power can be minimized only if the client selects a shortest path route toward the server. The state model noise w_n accounts for the fact that inter-departure times between consecutive packets from the client could not be constant, possibly due to operating system delay variations. The power of this noise process is fully controlled by the client and could be estimated by the client's own

timestamps $\{C_1^n\}$. Alternatively, we can treat that noise as additional measurement noise (v_n) and simply ignore it ($w_n = 0$). That was the approach followed in this work.

Assuming v_n a zero mean process and e^n, from (7) a stationary, nonzero mean process with uncorrelated consecutive samples, the following equation is derived:

$$E[v_i\, v_j] = \begin{cases} R & i = j \\ -R/2 & i = j+1 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$
$$R = variance(y_n),\ n \in 1..N.$$

Under the above assumptions and using vector notation, the measurement and state model equations become:

$$\begin{bmatrix} y_n \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} \delta t \\ \delta t \end{bmatrix} + \begin{bmatrix} v_n \\ v_{n-1} \end{bmatrix} \Rightarrow \tag{12}$$
$$\overline{\mathbf{y_n}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \delta t + \overline{\mathbf{v_n}}, \tag{13}$$
$$s_{n+1} = s_n = \delta t,\ n \in 1..N. \tag{14}$$

The Kalman filter "predict" and "update" equations are omitted and could be found in a relevant textbook [15]. The Kalman filtering technique is a recursive scheme; therefore, the estimate s_n converges to the correct value of δt after a number of messages $(C_1^n, t_2^n, t_3^n, C_4^n)$. The initial predicted value $s_{0|-1}$ was set to δT, and the associated error variance was set to R. After the Nth packet, the frequency of the client clock is obtained by the output $\hat{s} = s_{N|N}$ of the Kalman filter:

$$\hat{\phi} = \frac{\delta T}{\hat{s}} = \frac{\delta T}{\hat{\delta t}}. \tag{15}$$

From (7), averaging N measurements results in the following equation:

$$\frac{1}{N}\sum_{n=1}^{N} y_n = \delta t + \frac{1}{N}(\underbrace{e^2 - e^1}_{v_1} + \underbrace{e^3 - e^2}_{v_2} \cdots + \underbrace{e^{N+1} - e^N}_{v_N}), \tag{16}$$
$$\frac{1}{N}\sum_{n=1}^{N} y_n = \delta t + \frac{1}{N}(e^{N+1} - e^1). \tag{17}$$

The average value of N measurements could be used as a naive estimator of δt, and consecutively of clock rate via (15). The variance of this estimate, under the same assumptions for the noise process v_n, drops with N^2, as $var(e^{N+1} - e^1) = 2\ var(e_n) = var(v_n)$. Despite its attractive simplicity, this estimator provides large errors, compared to all the other approaches presented in this work, especially when a small number (N) of messages are used, as we will see in the following sections.

For the estimation of time offset θ, we could use (2). However, for a large number N of packets used, the duration of the experiment multiplied by the frequency skew could contribute to a significant synchronization error (e.g., 100 packets spaced 1 s from each other correspond to an additional time offset of 4 ms for a 40 ppm clock). Therefore, the estimate of the frequency offset should be exploited in the time offset calculation.

From Fig. 2 we have the following relationships:

$$C_1^n - \phi\, t_2^n = \theta - \phi\,(d_1 + q_1)^n \tag{18}$$
$$C_4^n - \phi\, t_3^n = \theta + \phi\,(d_2 + q_2)^n \tag{19}$$
$$\Downarrow$$
$$C_1^n - \phi\, t_2^n \le \theta - \phi\, d_1, \tag{20}$$
$$C_4^n - \phi\, t_3^n \ge \theta + \phi\, d_2. \tag{21}$$

Therefore, an estimate of θ is obtained by the following relationship:

$$\hat{\theta} = \frac{\max(C_1^i - \hat{\phi}\, t_2^i) + \min(C_4^j - \hat{\phi}\, t_3^j)}{2}. \tag{22}$$

Alternatively, Kalman filtering could be used again for the estimation of time offset θ. The estimate of clock rate ϕ from the above technique could be exploited to adjust the timestamps $C_1^n \leftarrow C_1^n/\hat{\phi}$, $C_4^n \leftarrow C_4^n/\hat{\phi}$ at the client side. Then measurements of time offset θ according to (2) could be filtered using standard, one-dimensional Kalman equations, with measurement model given by (4). The output estimate of θ after Kalman filtering of N measurements also is reported in the experimental results section.

B. Linear Programming (LP)

This line-fitting technique exploits both the forward and reverse path timestamps, by estimating a clock line that minimizes the distance between the line and the data, leaving all the data points below the line on a (t_2, C_1) plane or above the line on a (t_3, C_4) plane. The following equations describe the problem and its solution:

1. Forward Path:

$$\alpha_1 = \phi, \qquad \beta_1 = \theta - \phi\, d_1, \tag{23}$$
$$(20) \Rightarrow \alpha_1\, t_2^n + \beta_1 - C_1^n \ge 0,\ \forall\, n \in [1..N]. \tag{24}$$

Find α_1, β_1 that minimize:

$$f(\alpha_1, \beta_1) = \sum_{n=1}^{N} (\alpha_1\, t_2^n + \beta_1 - C_1^n), \tag{25}$$

under the constraint of (24).

2. Reverse Path:

$$\alpha_2 = \phi, \qquad \beta_2 = \theta + \phi\, d_2, \tag{26}$$
$$(21) \Rightarrow C_4^n - \alpha_2\, t_3^n - \beta_2 \ge 0,\ \forall\, n \in [1..N]. \tag{27}$$

Find α_2, β_2 that minimize:

$$f(\alpha_2, \beta_2) = \sum_{n=1}^{N} (C_4^n - \alpha_2\, t_3^n - \beta_2), \tag{28}$$

under the constraint of (27):

$$\hat{\phi} = \frac{\alpha_1 + \alpha_2}{2}, \tag{29}$$

$$\hat{\theta} = \frac{\beta_1 + \beta_2}{2}. \tag{30}$$

The simple and intuitive derivation above sets this technique as a strong candidate for clock parameter estimation.

C. Averaged Time Differences (ATD)

This method can be best described by Fig. 1. The time offset x_n is computed according to the NTP formula (2); therefore, this method has all the limitations discussed at the NTP section above. Differences of the time offset estimates provide estimates for the frequency offset. Particularly, clusters of 25–50 closely spaced messages are used, time offsets are computed, and the results are averaged to a single data point for the time offset. Then that is used in the following formula for frequency offset estimation:

$$\hat{f}(t_{n+1}) = \frac{x_{n+1} - x_n}{\tau}, \qquad \hat{f} \equiv \hat{\phi} - 1, \tag{31}$$

$$y(t_{n+1}) = \frac{y(t_n) + \alpha \hat{f}(t_{n+1})}{1 + \alpha}. \tag{32}$$

The value of τ nominally should be equal to $t_{n+1} - t_n$; however, this quantity cannot be measured by the client's own clock. Nevertheless, for small values of the frequency offset, this can be set to $C(t_{n+1}) - C(t_n)$ because that is what the client can measure. The estimated frequency offset is averaged again using an exponential filter with a time constant α that depends on the stability of the local oscillator. Then the filtered frequency offset is used in the following formula, which also is depicted in Fig. 1.

$$\hat{x}(t_{n+1}) = \hat{x}(t_n) + y(t_n)(\tau). \tag{33}$$

A variant of this method is used in this work. Frequency offsets are calculated using (31), then filtered using the above exponential filter with $\alpha = 0.5$. The final frequency offset estimation is the mean of all the N exponentially filtered frequency offsets calculated at each epoch.

The power of this method is its simplicity. For the Gaussian case in which consecutive measurements are independent from each other, an increase of samples averaged by a factor of N reduces the variance of the estimate by a factor of N. Therefore, there is a trade-off between accuracy achieved and the cost of realizing it.

IV. Performance

In this section we evaluate the performance of the three algorithms in two separate cases:

- The Gaussian case in which the queuing delay difference between two consecutive NTP messages is a Gaussian random variable. Consequently, the dispersion of the packets at the server is also a Gaussian random variable. In this experiment, consecutive measurements are independent.
- The self-similar case in which multiple pareto connections aggregate and form cross-traffic with long-range dependence.

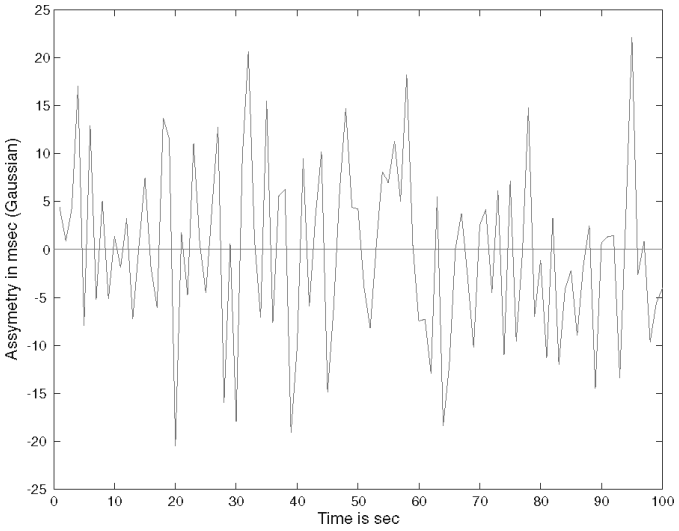

Fig. 3. Asymmetry of delays between forward (to server) and reverse (to client) path for Gaussian case.

The estimate, the variance of the estimate, and the number N of packets used at each epoch are reported. In both cases the true clock frequency offset $\phi - 1$ was +40 ppm, and the time offset θ was 20 ms. The NTP messages were transmitted at intervals of 1000 ms. Each experiment was run 300 times.

A. The Gaussian Case

In Fig. 3 we present the asymmetry between forward (to server) and reverse (to client) path, from a sample run. The average round-trip time was on the order of 40 ms, consecutive measurements were independent and identically distributed. In Figs. 4 and 5 we present the average estimate and the standard deviation of the estimate for the frequency offset $\phi - 1$ and time offset θ, respectively, as a function of number N of packets used.

The Kalman filter performed better when the number of packets N was above the minimum number of samples needed for convergence (on the order of 25–30 packets). This experimental finding is validated by the fact that the Kalman filter (at steady-state) is the optimal linear estimator in the presence of Gaussian noise. The LP technique performed better than both averaging techniques (ATD and naive estimator), which performed well only if a large number of messages were used.

Frequency offset estimate variance was decreased with number N of packets used. From Fig. 4 it is shown that the standard deviation of the estimate drops linearly with N (variance drops with N^2) for Kalman filtering as well as for

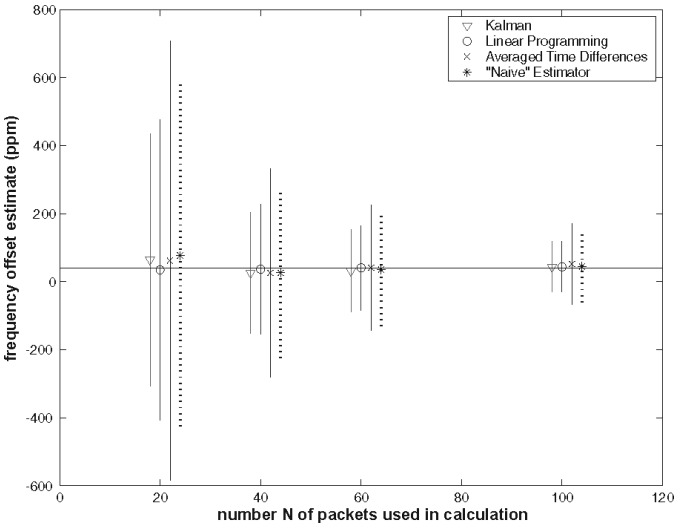

Fig. 4. Frequency offset estimate and standard deviation as a function of N (number of packets used), for Gaussian case.

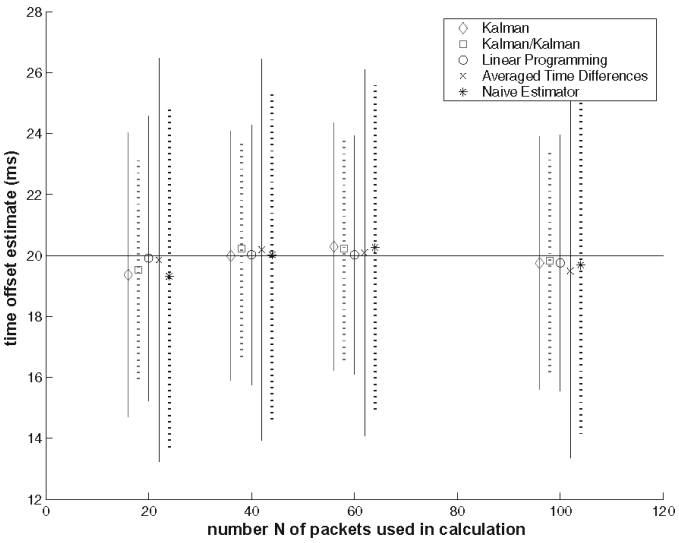

Fig. 5. Time offset estimate and standard deviation as a function of N (number of packets used), for Gaussian case.

the naive estimator, as expected, and the variance as well as the error is smaller for the Kalman algorithm. Time offset estimates were close to the real value, regardless of N. This can be justified by the fact that the algorithms presented here focus on the accurate calculation of frequency offset that was set at 40 ppm in this experiment. Error in the calculation of a 40 ppm quantity over a duration of 100 s (1 packet every 1000 ms) is negligible in the calculation of time offset (using the algorithms described above)[3] and, of course, not visible at the time scales of Fig. 5.

[3]Time offset θ was estimated using the same algorithm for Kalman, ATD, and naive, described in the Kalman filtering section. Kalman filtering for both time and frequency offset is depicted as Kalman/Kalman.

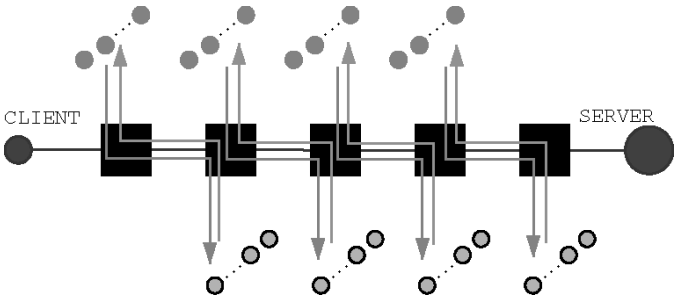

Fig. 6. Simulation in ns-2 with pareto cross traffic; 14 connections per link per direction.

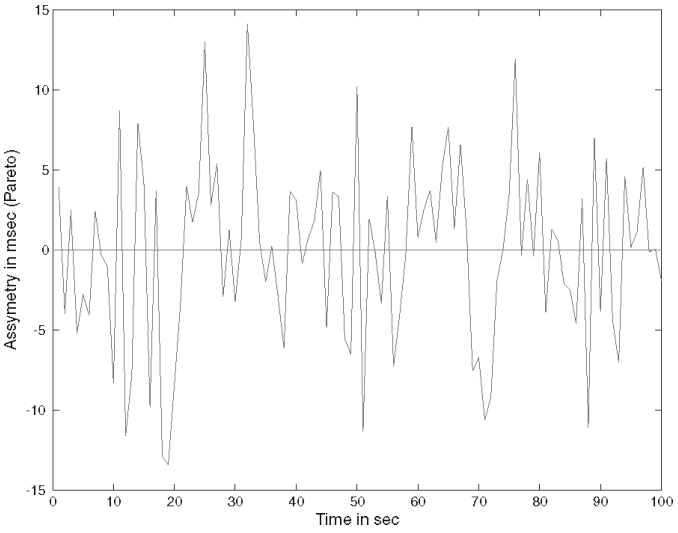

Fig. 7. Asymmetry between forward and reverse path with self-similar traffic.

B. The Self-Similar Case

In this section, we are investigating the performance of the three algorithms in the presence of bursty traffic. It has been shown that the aggregation of many on/off sources could form a self-similar source, exhibiting long-range dependence [16]. The fact that local area network traffic demonstrates chaotic (self-similar) behavior [17] motivates the test of the three algorithms in a self-similar environment that is fundamentally different from the Gaussian case for which Kalman filtering seems appropriate.

Fig. 6 displays the simulation setup in network simulator 2 (ns-2) [18]. The use of the links was 90%, the average round-trip time was on the order of 40 ms, and the asymmetry between the forward and reverse path is depicted in Fig. 7. The inter-departure time of the NTP packets remains 1000 ms.

Figs. 8 and 9 show in a sample run how well the Kalman filter locks onto the correct interarrival time δt and frequency offset value $(\phi - 1)$. Fig. 10 shows how well the ATD technique (with the exponential filter) locks onto the frequency offset value $(\phi - 1)$. The internal line is the filtered waveform through a low pass filter. Fig. 11 displays the one-way delay across the reverse path as a function

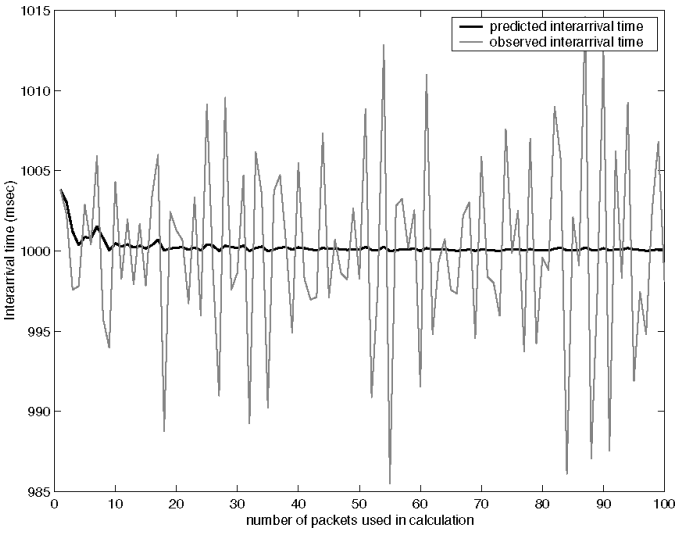

Fig. 8. Predicted interarrival and measured interarrival interval using the Kalman filter for self-similar cross traffic.

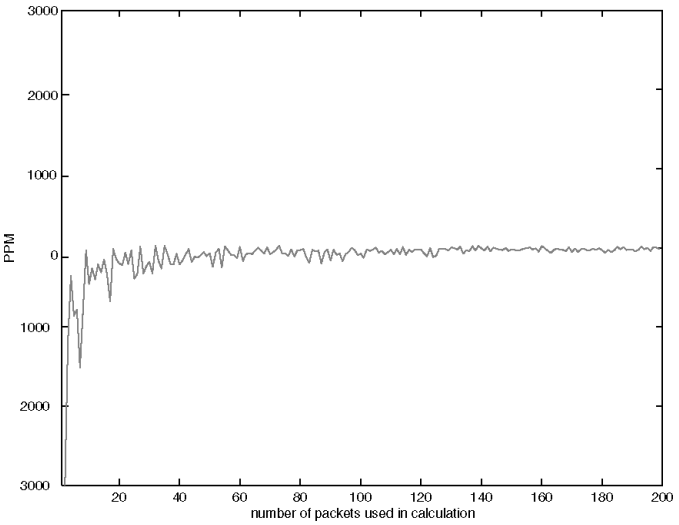

Fig. 9. Estimation of frequency offset $\phi - 1$ using the Kalman filter for self-similar cross traffic.

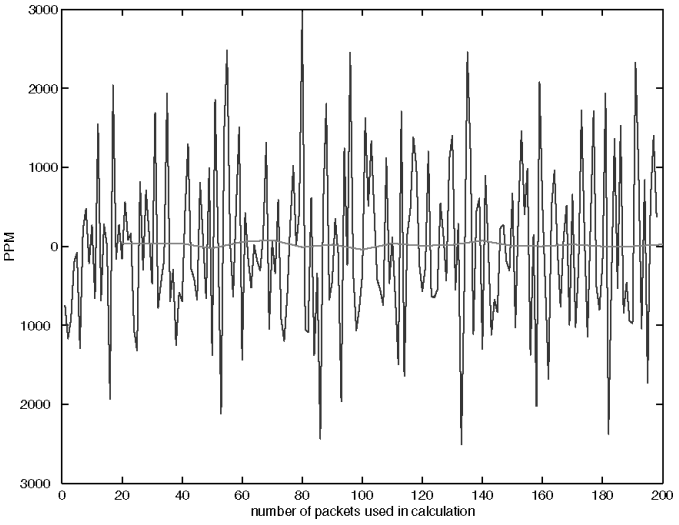

Fig. 10. Estimation of frequency offset $\phi - 1$ using the ATD technique. Low pass filtering of data is also plotted.

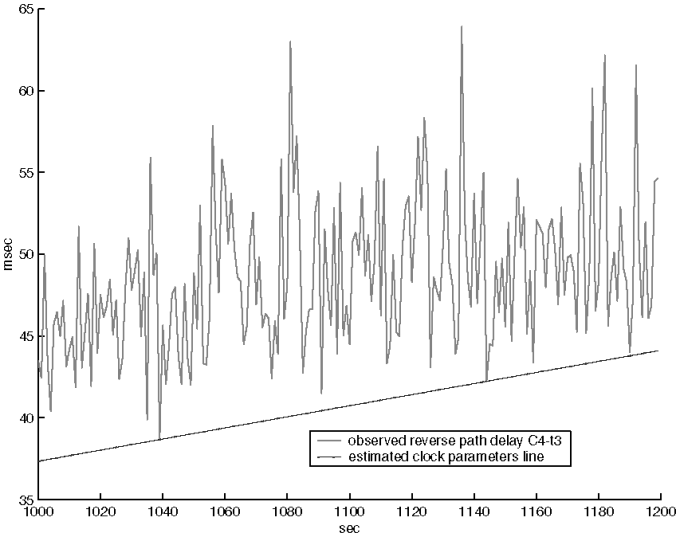

Fig. 11. Delay $C_4^n - t_3^n$ from the reverse path and clock line estimation using LP for self-similar cross traffic.

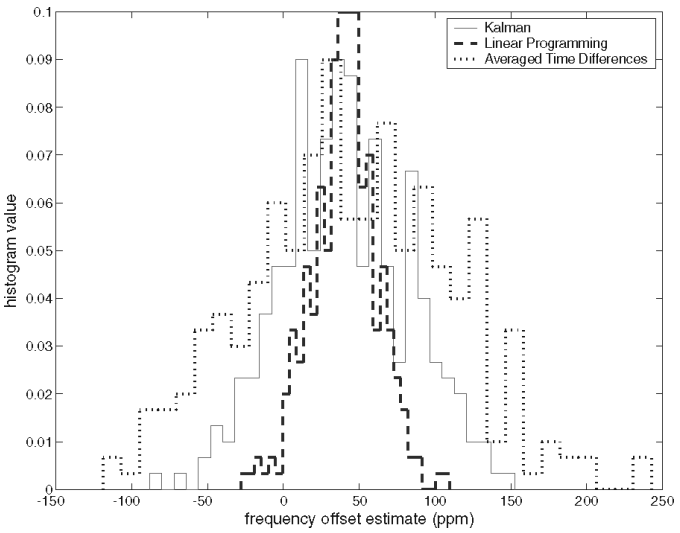

Fig. 12. Histogram of the frequency offset estimates for self-similar cross traffic.

of time. The trend of the plot is coherent with the following derivation. The clock line $\hat{\phi}\, t_3^n + \hat{\theta}$ with parameters estimated by the LP technique also is depicted:

$$\begin{aligned} &(21) \Rightarrow \\ &C_4^n - t_3^n = (\phi - 1)\, t_3^n + \phi\, (d_2^n + q_2^n) + \theta \Rightarrow \\ &C_4^n - t_3^n \geq (\phi - 1)\, t_3^n + \phi\, d_2^n + \theta. \end{aligned} \tag{34}$$

Fig. 12 shows the histogram of frequency offset estimates for the self-similar case, for $N = 100$, and Fig. 13 shows the performance of the three algorithms in the estimation of frequency offset, for various number N of packets used in the calculation. Time offset estimation $\hat{\theta}$ resulted in significant errors due to asymmetry between forward and reverse path, as expected (Fig. 14).

From the above diagrams, it is deduced that the Kalman filtering technique no longer produces the best estimates

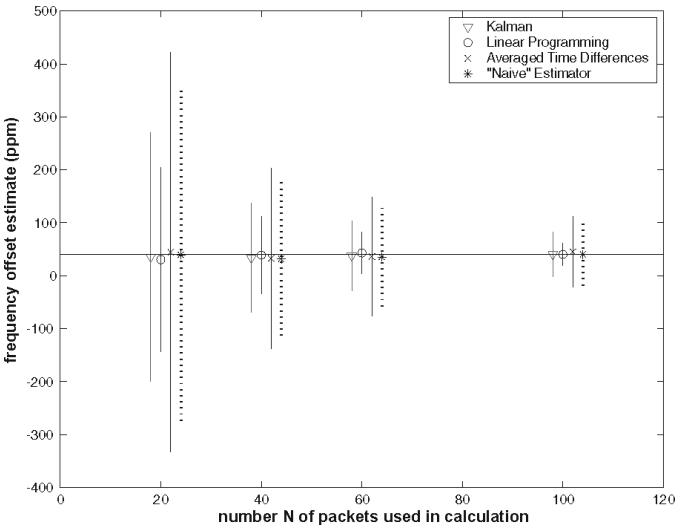

Fig. 13. Frequency offset estimate and standard deviation for self-similar cross traffic as a function of number N of packets used in calculation.

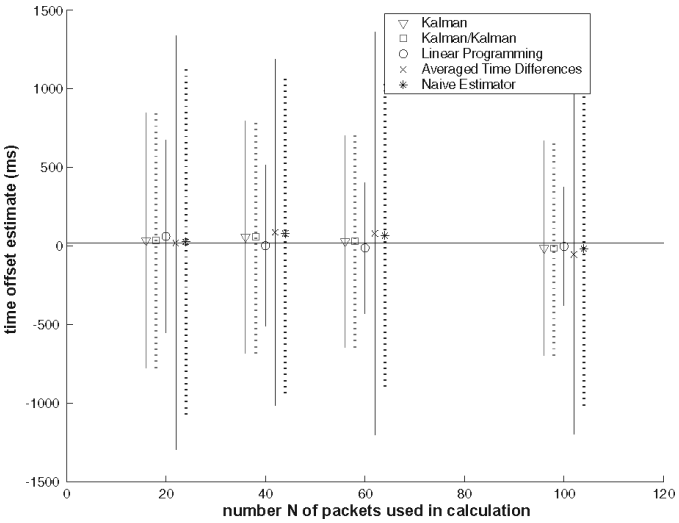

Fig. 14. Time offset estimate and standard deviation as a function of N (number of packets used), for self-similar traffic.

with the smallest variance. The noise is no longer Gaussian, so Kalman filtering is not optimal and LP performs better in the presence of bursty traffic both in terms of estimation error (accuracy) and its variance (precision). For the same reason (burstiness and asymptotically long-range dependence as opposed to the Gaussian distribution around the mean), ATD and naive estimator perform inferiorly than the LP technique. All algorithms for frequency offset estimation reduce the standard deviation (and therefore variance) of the estimate with increased number N of packets used, and the relation between that improvement and N seems linear (therefore, variance drops with N^2), as can be seen in Fig. 13.

TABLE I
FREQUENCY OFFSET ESTIMATION USING AN EXISTING NTP/GPS SERVER.

	Kalman	LP	ATD
$\widehat{\phi} - 1$ (PPM)	101.6	54.7	122.7

C. Measurements

In order to emphasize the end-to-end character of the algorithms evaluated (especially for the case of Kalman filtering and LP), we modified NTP client daemon and exchanged 100 packets at intervals of 1 s with a stratum-0 server (connected to GPS). The time server was geographically located at Palo Alto, CA, 3100 miles away from our client machine, with average round-trip time 85 ms, 18 hops away. We then processed the packets according to the algorithms evaluated above, and the frequency offset estimation results are presented at Table I.

An interesting idea could be averaging the two estimates calculated according to Kalman and LP because the former performed better at the Gaussian case and the latter at the self-similar one.

V. CONCLUSIONS

The Kalman filtering technique, optimal for the Gaussian case, needed a considerable number of packets in order to converge (on the order of 20–30 packets for the formulation adopted and the experimental setup). Nevertheless, the technique performed well at the self-similar case as well, with improved performance in terms of error and variance of the estimate when the number of packets N increased. The algorithm estimates the variance of network delay (jitter) and uses that estimate to calculate the frequency and time offset model variables. The algorithm can be applied without major operation requirements in the NTP-client daemon and requires no modifications in the NTP-server daemon. It could benefit by scheduled transmission from the client system that ensures minimum delay variance due to the operating system.

The linear prediction technique surpassed all the other techniques at the case of bursty traffic approximating real-word, long-range dependence (chaotic) conditions, even though it had inferior performance when measurements were completely independent. Its intuitive structure makes it attractive for straightforward implementation.

Averaging as exploited and implemented in the averaging time differences technique (where equal intervals between measurements were used, and, therefore, averaging differences of time were equivalent to averaging frequency offset estimates) performed inferiorly to the LP and Kalman filtering techniques at the self-similar case in which measurements are not independent. However, its simplicity makes it attractive, especially at the cases in which a small number of measurements are available or

a trade-off between accuracy and the cost of realizing it cannot be avoided.

All three techniques showed improvement in terms of frequency offset estimation error and variance of the estimate with an increasing number of packets N. For Kalman filtering, the relationship between improvement and N seems linear (for standard deviation of frequency offset estimate) or quadratic (for variance of frequency offset estimate) and, therefore, increased accuracy is expensive in terms of number of packets used (communication bandwidth). This work tried to quantify that cost, by comparatively evaluating a number of diverse techniques.

Acknowledgment

The author would like to thank Thucydides (Duke) Xanthopoulos for his help throughout this work. He is also grateful to the anonymous reviewers for their useful comments that improved the quality of the final document.

References

[1] J. Paradiso, K. Hsiao, J. Strickon, and P. Rice, "New sensor and music systems for large interactive surfaces," in *Proc. Int. Comput. Music Conf.*, Aug. 2000, pp. 277–280.

[2] G. Opshaug and P. Enge, "GPS and UWB for indoor navigation," presented at Institute of Navigation GPS Conf., Salt Lake City, UT, Sep. 2001.

[3] C. Parsa and J. J. Garcia-Luna-Aceves, "Improving TCP congestion control over internets with heterogeneous transmission media," *Proc. IEEE Int. Conf. Network Protocols*, Oct. 1999, pp. 213–221.

[4] D. L. Mills, "Network time protocol (version 3) specification, implementation and analysis," Request for Comments (RFC) 1305, Univ. Delaware, Mar. 1992.

[5] J. Levine, "Time synchronization using the internet," *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, vol. 45, no. 2, pp. 450–460, Mar. 1998.

[6] J. A. Barnes, A. R. Chi, L. S. Cutler, D. J. Healey, D. B. Leeson, T. E. McGunigal, J. A. Mullen, Jr., W. L. Smith, R. L. Sydnor, R. F. C. Vessot, and G. M. R. Winkler, "Characterization of frequency stability," *IEEE Trans. Instrum. Meas.*, vol. IM-20, pp. 105–120, May 1971.

[7] J. Levine, "Introduction to time and frequency metrology," *Rev. Sci. Instrum.*, vol. 70, no. 6, pp. 2567–2596, Jun. 1996.

[8] J. Levine, Personal communication, National Institute of Standards and Technology, Mar. 2002.

[9] V. Paxson, "Measurement and analysis of end-to-end internet dynamics," Ph.D. dissertation, University of California at Berkeley, Apr. 1997.

[10] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," in *Proc. 18th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, 1999, pp. 227–234.

[11] J. Levine, "An algorithm to synchronize the time of a computer to universal time," *IEEE/ACM Trans. Networking*, vol. 3, no. 1, pp. 42–50, Feb. 1995.

[12] G. D. Troxel, "Time surveying: Clock synchronization over packet networks," Ph.D. dissertation, Massachusetts Institute of Technology, May 1994.

[13] A. Bletsas, "Time keeping in myriad networks: Theories, solutions and applications," M.S. thesis, Massachusetts Institute of Technology, Jun. 2001.

[14] N. Gershenfeld, *The Nature of Mathematical Modeling*. Cambridge, UK: Cambridge Univ. Press, 2000.

[15] S. J. Orfanidis, *Optimum signal processing, an introduction*. 2nd ed. New York: McGraw-Hill, 1988.

[16] M. S. Taqqu, W. Willinger, and R. Sherman, "Proof of a fundamental result in self-similar traffic modeling," in *ACM Comput. Commun. Rev.*. vol. 27, no. 2, pp. 5–23, Apr. 1997.

[17] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," *IEEE/ACM Trans. Networking*, vol. 23, no. 4, pp. 183–193, Oct. 1993.

[18] The Network Simulator-ns-2, http://www.isi.edu/nsnam/ns/.

Aggelos Bletsas (S'98–A'99) received his diploma degree (with excellence) in Electrical and Computer Engineering from Aristotle University of Thessaloniki, Greece in 1998. He joined the Massachusetts Institute of Technology (MIT) Media Laboratory, Cambridge, MA, in 1999 where he earned the M.S. and Ph.D. degrees in June 2001 and September 2005, respectively, researching in the areas of Wireless Communication and Networking.

His research interests are in time/frequency metrology, scalable wireless communication and networking, with emphasis on relay techniques, signal processing and radio hardware/software implementations for wireless transceivers and low cost sensor networks.

He received best thesis award in 1999 from Ericsson and held a British Telecom Fellowship from 2000–2002 and a Nortel Networks Fellowship from 2002 to 2005, during his graduate studies at MIT.