Analysis of Computer Clock Characteristics for Location Determination

Neha Gupta
Department of Computer Science, UMD
College Park-20742

Abstract

One of the key pieces of information required for context-aware or ubiquitous computing is location. Knowing where a mobile device is located helps to provide a lot of information to the user. For example a context aware mobile phone may automatically reject unimportant calls if the user is with family or in a meeting. The user need not reject the unimportant calls manually, the phone knows that the user's location and may use that information to reject the unimportant calls. Few very common examples where location determination systems are useful are determining the location of a nearest printer, staircase, eating joint etc. Other scenarios where such systems are useful are Asset Tracking in Warehouses, Emergency Operations, and Social Networking etc. Pinpoint Technology[1] patented by the University of Maryland is a location determination technology that uses RF (Radio Frequency) propagation delays to determine the distance between two nodes and eventually the spatial location of the nodes. It is a TOA (Time of Arrival) based method and doesn't require synchronized clocks. Our main goal in this paper is to analyze the computer clock characteristics and try to estimate the variance in these characteristics for accurate location determination.

Related Work

This area has been a topic of active research. One of the most ubiquitous location determination device used at present times is the Global Positioning system (GPS)[2]. It tells the physical location of a device "latitude and longitude" using a system of satellites which hover around the earth. The clocks of the GPS receivers are synchronized with highly accurate atomic clocks in the GPS satellites with 1 microsecond accuracy. GPS uses trilateration and calculates the time of flight of signals sent by atleast four satellites to calculate the location of the receiver in a three-dimensional space. But the main drawback of the GPS technique is that it cannot be used indoors since satellite signals are not received inside buildings.

Another technique, which is most often used, is the "echoing technique" [3][4]. In order to estimate the propagation delay between two nodes, the round trip time (RTT) is measured. The advantage of this type of system is that since both the measurements are taken at the same end clock synchronization is not required. The main disadvantage of this system is that it requires require $O(n^2)$ message exchanges. It is computationally expensive.

One of the other techniques used is RSSI (Receiver Signal Strength Indicator) based method [5].This technique uses the magnitude of the receiver signal strength to measure the location of an object. It consists of two phases – Offline phase and online phase. In Offline phase, an area (a building or mall) is calibrated by creating a radiomap of the signal strengths at selected locations. During the online phase the received signal strength is compared with the calibrated signal strength to determine the most probable location of a node. This type of method is not very useful because of the initial calibration is required, so location cannot be determined on adhoc basis. Secondly, it takes many man-hours to calibrate a huge building or mall which is a big bottleneck.

Angle of Arrival [6] techniques can also be used but they require a directional antenna to measure the angle at which a wireless signal is received. The directional antennas are expensive and difficult to maintain.

Introduction

Computer clocks in a network are rarely synchronized. Resetting them to a highly accurate time source is rendered useless by clock skew and drift. Many applications like network congestion control tools, location determination modules and distributed systems have time based events which rely on accurate time measurements made across end hosts. The lack of synchronization of the end systems' clocks reduces the accuracy of these measurements.

The basic clock model that we start with is

- The local clock time τ_A of machine A is related to the local time of machine B τ_B by the formula below :

$$\tau_A(t) = \beta\ (\tau_B(t) + \alpha)$$

- We assume that
- α and β are the relative offset and relative skew of the clock and are assumed constant for short periods.
- τ_A and τ_B are the local clock times at node A and B respectively.
- t is the real time, which is unknown.

- **Offset:** The offset of clock A relative to clock B at time t is α, which is simply the difference in the time reported by clock A and B
- **Skew:** The skew of clock A relative to clock B at time t is β , which is the difference in the frequencies of the two clocks

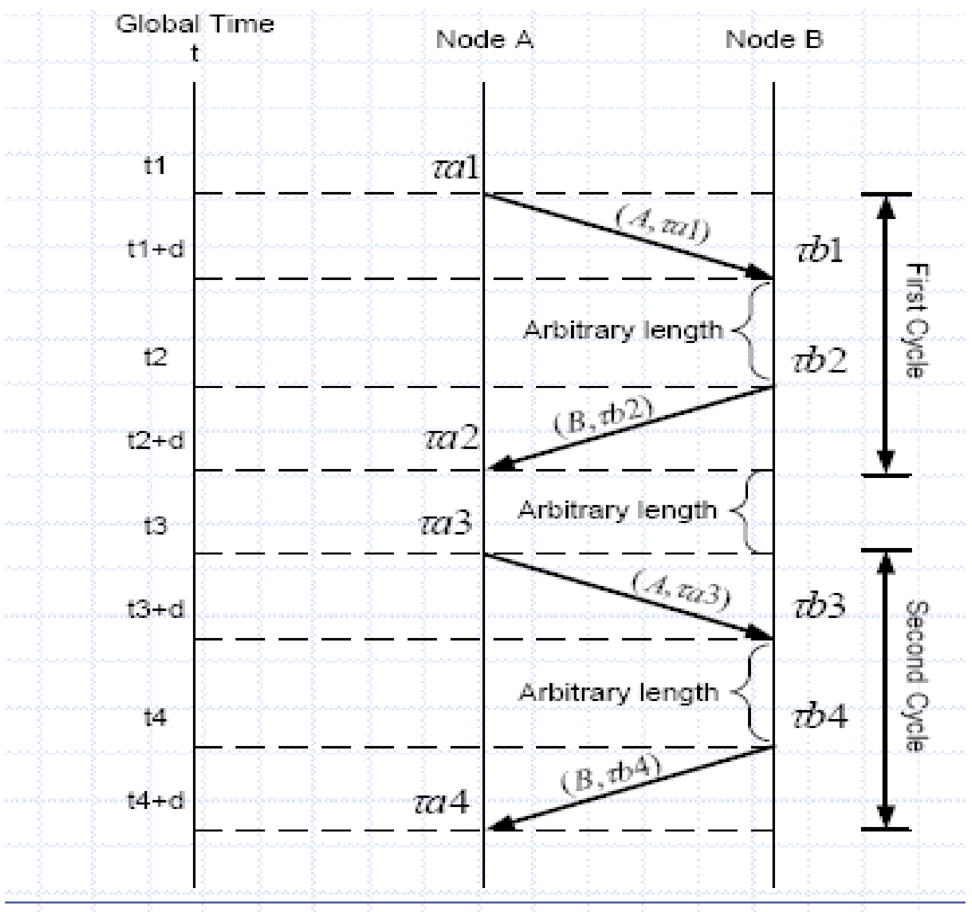


Fig 1: Basic diagram of pinpoint algorithm.

The Pinpoint[1] algorithm , which this paper analysis, consists of three phases :

- **Measurement Phase**: Each node transmits a message containing its ID and the transmit(send) timestamp of the message, and records the receive timestamp of the messages sent by other nodes

  - In the first measurement cycle, node A broadcasts, at global time t1, a message containing its id and send timestamp (A, τ_a (t1)).

$$\tau_{a1} = \beta_a (\alpha_a + t_1)$$

  - Node B receives it at global time t_1+d and records the receive timestamp as equaling

$$\tau_{b1} = \beta_b (\alpha_b + t_1 + d)$$

  - This process is repeated in the second measurement cycle as shown in fig1.

- ❑ **Information Exchange Phase**: After the measurement phase, each node transmits a message containing its receive timestamp for messages transmitted by other nodes in a measurement phase and both nodes have the following set of equations.

$$\tau_{a1} = \beta_a(\alpha_a + t_1) \qquad \tau_{b1} = \beta_b(\alpha_b + t_1 + d)$$

$$\tau_{a3} = \beta_a(\alpha_a + t_3) \qquad \tau_{b3} = \beta_b(\alpha_b + t_3 + d)$$

$$\tau_{a2} = \beta_a(\alpha_a + t_2 + d) \qquad \tau_{b2} = \beta_b(\alpha_b + t_2)$$

$$\tau_{a4} = \beta_a(\alpha_a + t_4 + d) \qquad \tau_{b4} = \beta_b(\alpha_b + t_4)$$

- ❑ **Computation Phase** :

Each node is able to calculate relative skew and propagation delay.

- Relative skew

$$\frac{\tau_{a3} - \tau_{a1}}{\tau_{b3} - \tau_{b1}} = \frac{\beta_a(\alpha_a + t_3) - \beta_a(\alpha_a + t_1)}{\beta_b(\alpha_b + t_3 + d) - \beta_b(\alpha_b + t_1 + d)} = \frac{\beta_a}{\beta_b}$$

- Propagation Delay

$$\beta_b d = \frac{(\tau_{b1} - \tau_{a1}) + (\tau_{a2} - \tau_{b2})}{2} + \frac{1}{2}\left(\frac{\beta_a}{\beta_b} - 1\right)(\tau_{a2} - \tau_{a1})$$

Hypothesis

In pinpoint we assume the clock skew to be constant with time. But the question we try to raise in this paper is –Do the clocks have constant skew or is there variability in the skew? Can we estimate how much is the variability of the skew? What can be the possible reasons for the variance?

Experimental Setup

For our experiment we decided to monitor to characteristics of access point clock. We measure time of flight characteristics of the wireless signals. The parameters which we vary in our experiments are

- **Sender side clocks (Access Points)**: 802.11 Access Points transmit beacon messages which are called the "heartbeat" messages every 100 ms to establish communication with the NIC (Network Interface cards) which are within range of the Access Point .These beacon frames contain a 64 bit value of the sending timestamp. To determine typical network characteristics we need to perform experiments for multiple access points. We used Cisco Access

Points and Zcomax Access Points deployed by the Office of Information Technology at the University.

- **Receiver side clocks (Network Monitoring Devices):** There are a number of Wireless Network Monitoring tools available commercially as well as free. We use Cace Technologies Network cards[7] on Windows XP as well as Netgear Network Cards with Wireshark[8] on Suse Linux Distribution. Both the above cards use radiotap headers to receive timestamp information.

- **Time of the experiment:** We intend to perform the experiment over different lengths of time, to see the variability in the clock characteristics over time. We plan to record timestamps for a large time period and do a binary search over this time-period to divide it into shorter time-intervals. Then comparing the variability in the skew over different intervals will give us a good measure of the clock behavior.
-

Dependent Variables

- Send time
- Receive time
- Sequence number

.

Control Variables

- The location of the sender and the receiver is a controlled variable and we make both the sender and the receiver static during the experiment.

Experimental Protocol

For timestamp measurements we use the following experimental protocol,
1) We perform the experiment using two laptops one enabled with CACE Network Drivers on Windows XP machine and the other enabled with Wireshark with Netgear WG511T card on a Linux machine.
2) We monitor the network and decide an access point to monitor.
3) The capture programs of both the laptops are set to capture just the beacon frames of the access point chosen in Step 2). (All other capture information is useless.)
4) The captured beacon frames are parsed to get the send timestamps, receive timestamps and the sequence numbers.

Two different capture programs are used in the experiment in order to check if the variability in the clock characteristics was due to the access point clocks or due to the receiver side variability.

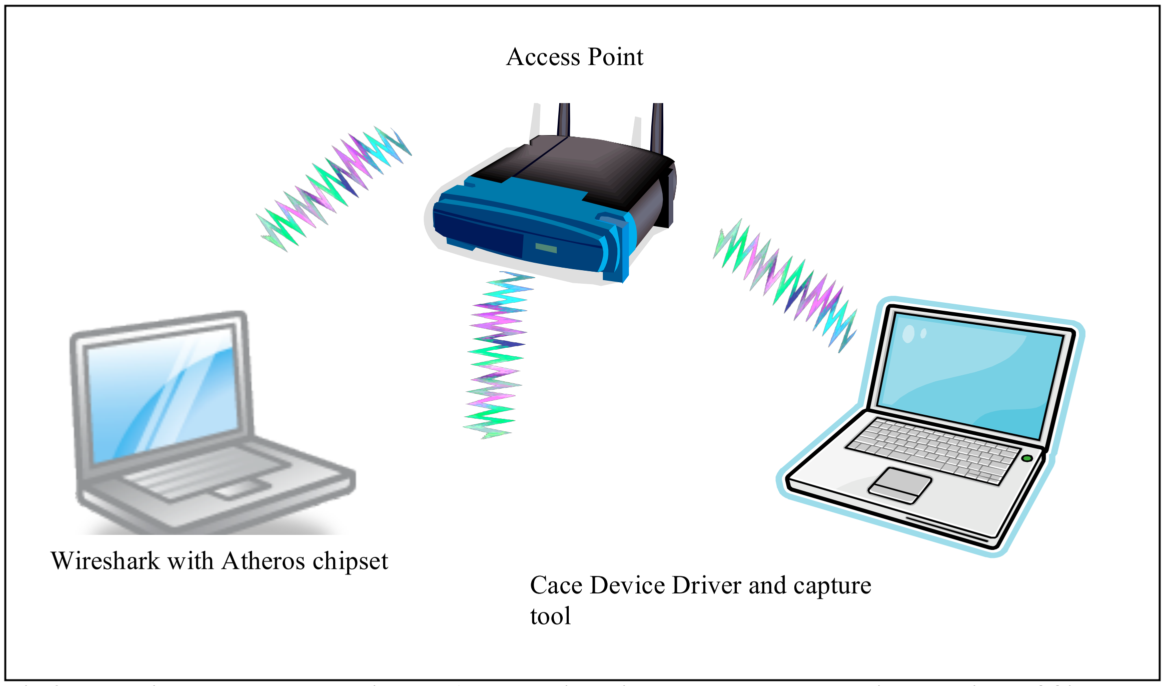

Access Point
Wireshark with Atheros chipset
Cace Device Driver and capture tool

Fig 2: Experimental setup showing an Access Point with two laptops recording the time of flight measurements for the beacon packets.

Experimental Measurements and Analysis

Beacon messages were parsed at the receiver and the sequence numbers, send timestamps and the receive timestamps were obtained.

The calculated values are:

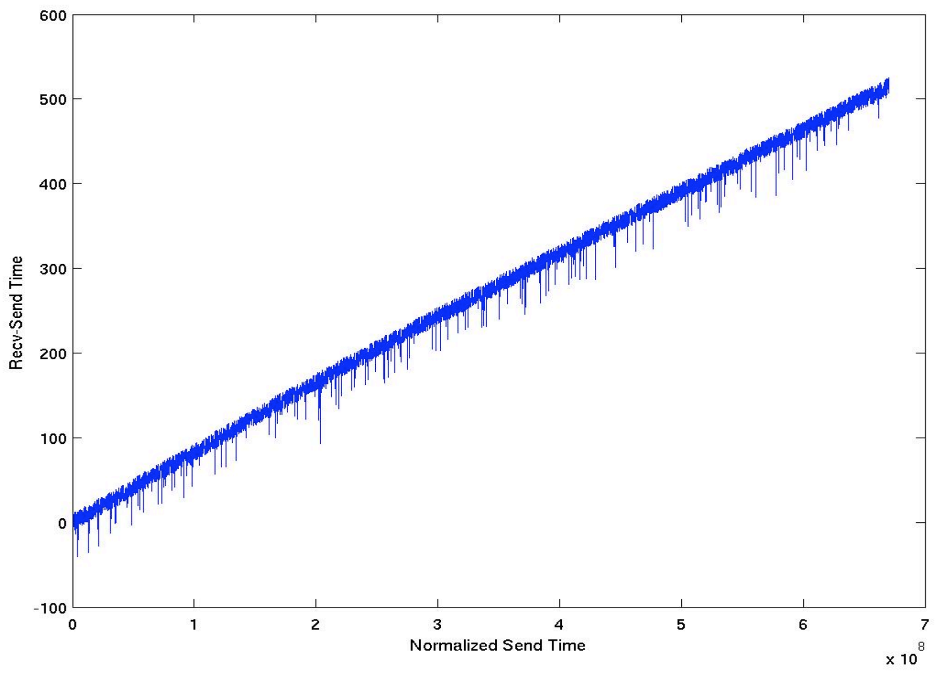
- ❑ Skew
- ❑ Deviation of time measurements from linear estimated time model.

$$\tau_A'(t) - \tau_B'(t) = (\beta - 1)\, \tau_B'(t) + \varepsilon$$

- ■ $\tau_A'(t)$, $\tau_B'(t)$ are the normalized send/receive timestamps. We use the normalize time stamps to remove the offset.
- ■ β is the relative skew between the two clocks.
- ■ ε is the noise in the measurements with mean 0 and variance, which we calculate from β estimation.

We performed the capture of a single access point (MAC 00:14:f1:ad:41:20) simultaneously using the CACE adapter on a Windows laptop and Wireshark using the madwifi drivers on a Linux machine. The duration of the capture was approximately 10 minutes. Measurements were also taken over longer time periods, but this short capture was sufficient for illustration of problems with the CACE capture.

We plotted the difference of normalize receive and send time versus the normalized send time for both Cace and Wireshark readings and found that for the same set of beacon signals the Cace readings showed a lot of deviation(variability) from the linear model. Fig 3 and Fig 4 represent the graphs obtained from Cace and Wireshark respectively.



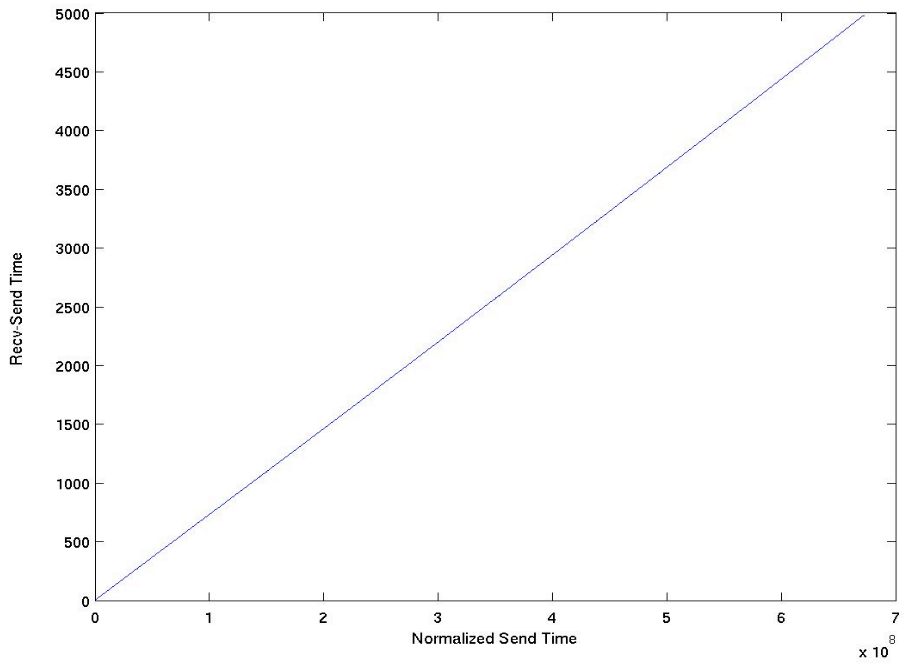


Fig 4: Wireshark Measurements with (Receive – Send) time vs. the normalized send time

To further verify that Cace Measurements were problematic we calculated the standard deviation of the difference of the estimated time of flight and measured time of flight for both Cace and Wireshark readings .We estimated the β using the linear regression slope for the first 300 readings assuming β to remain constant for short durations. The readings that we obtained were Fig 5 showed that there was a lot of deviation from linear in Cace Readings as compared to the Wireshark readings.

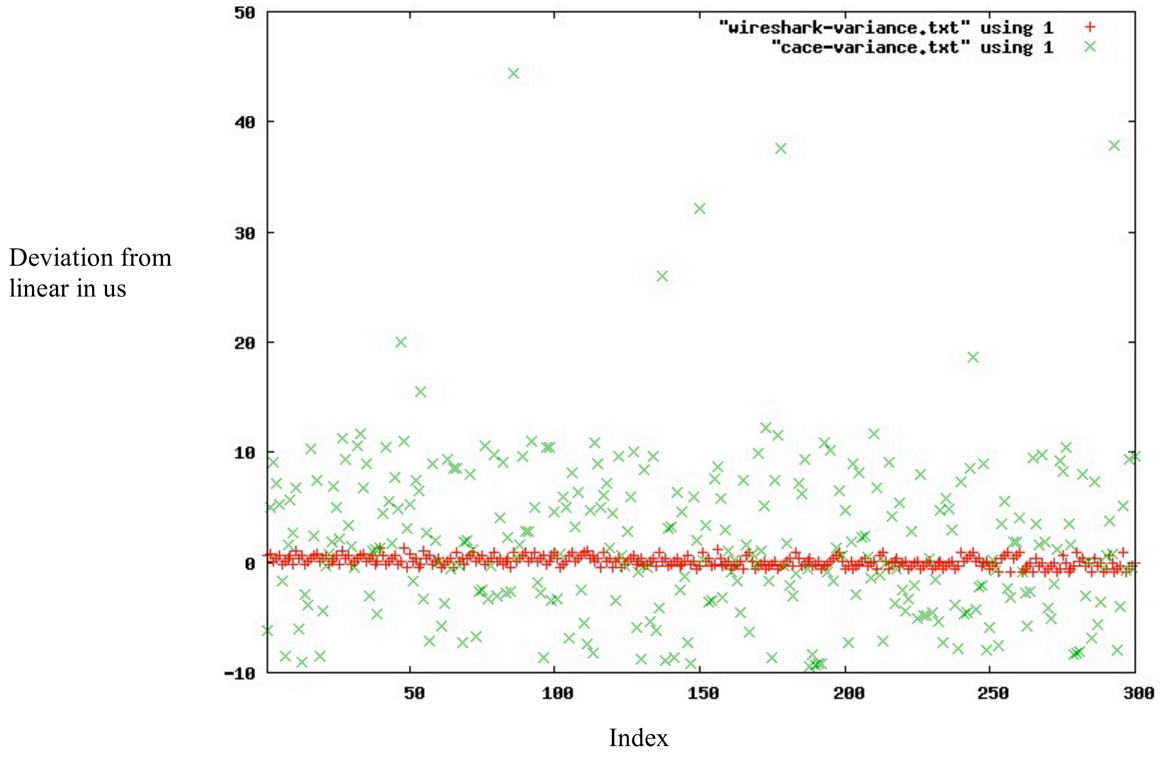


Fig 5: Cace and Wireshark Measurements with deviation from linear on the y axis versus the index number on x-axis

The other problems that we observed for Cace were, there were clear outliers for the send timestamps reported by the CACE adapter. The madwifi capture for the time period did not have any clear outliers. Two of the outliers are recorded with sequence numbers in order. The madwifi capture did not record, these two beacons. For the third outlier, the CACE capture recorded a single out-of-order sequence number of 2291. The madwifi capture recorded a sequence number of 3892 which is correctly ordered, with a timestamp in the expected range as well.
The timestamp values are shown in Tables 1 and 2.

CACE			Madwifi		
Receive	Send	Seq #	Receive	Send	Seq #
1137647745	5284329369841	2331	3891066834386035655	5284329369841	2331
1137750144	5284329472241	2334	3891066834386138056	5284329472241	2334
1137852543	5284329574640	2337	3891066834386240456	5284329574640	2337
1137954933	9223377321184452848	2340			
1138057798	5284329779891	2343	3891066834386445709	5284329779891	2343
1138159746	5284329881842	2346	3891066834386650059	5284329984240	2349
1138262127	5284329984240	2349	3891066834386752460	5284330086640	2352
1156284599	5284348006709	2993	3891066834404672664	5284348006709	2993
1156386923	5284348109040	2996	3891066834404774995	5284348109040	2996
1156591715	5284348313841	3002	3891066834404979797	5284348313841	3002
1156694124	163260770840593548	3005			
1156796681	5284348518789	3008	3891066834405184747	5284348518789	3008
1156898915	5284348621040	3014	3891066834405286999	5284348621040	3014
1157001318	5284348723441	3017	3891066834405389400	5284348723441	3017

Table 1: Outliers with in-order sequence number

CACE			Madwifi		
Receive	Send	Seq #	Receive	Send	Seq #
1182396510	5284374118644	3883	3891066834430784794	5284374118644	3883
1182498896	5284374221041	3886	3891066834430887192	5284374221041	3886
1182601304	5284374323441	3889	3891066834430989593	5284374323441	3889
1182703714	3315621548932724835	2291	3891066834431091994	5284374425841	3892
1182806098	5284374528241	3895	3891066834431194394	5284374528241	3895
1182908496	5284374630641	3898	3891066834431296796	5284374630641	3898
1183010910	5284374733041	3901	3891066834431399196	5284374733041	3901

Table 2: Outlier with out-of-order sequence number

The other problems with the Cace Capture were that a few initial readings were erroneous. After normalization the readings of the send and receive timestamps should be roughly the same since β is of the order of few parts per million, but for the Cace readings we got a huge difference in the first few readings of the send and the receive timestamps.

i	$t_{s,i}$	$t_{s,i} - t_{s,i-1}$	$t_{r,i}$	$t_{r,i} - t_{r,i-1}$
1	0	-	0	-
2	102400	102400	667	667
3	2764802	2662402	47940	47273
4	2867200	102398	150335	102395

Table 3: CACE Timestamps at startup

Results

- There were a number of deficiencies associated with timestamp measurements using Cace Device Drivers.
  - ❑ Presence of outliers
  - ❑ High deviation from linear
  - ❑ Presence of erroneous initial readings.

The variability in β

After concluding that Cace device drivers produced variability in the readings of the access point, we used Wireshark capture program to study the behavior of the clock characteristics of the access points.
We calculated the (skew-1) for cisco access points for a period of two hours (Fig 6) and observed that the variability in the skew was of the order of 1 ppm.

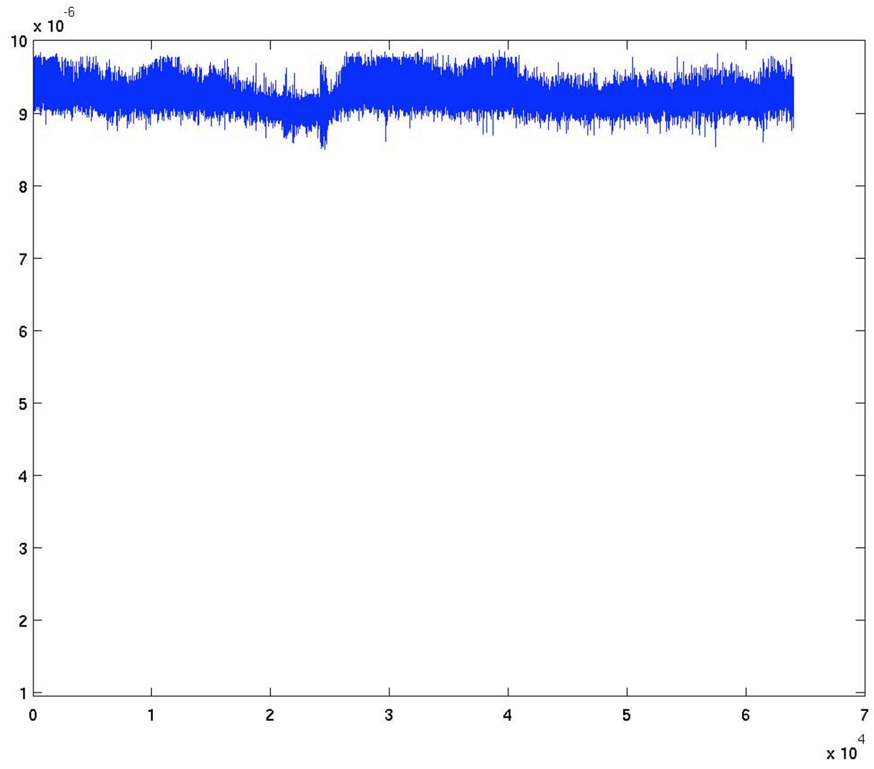


Fig 6: (β-1) plot for Cisco Access Points for a period of 2 hours

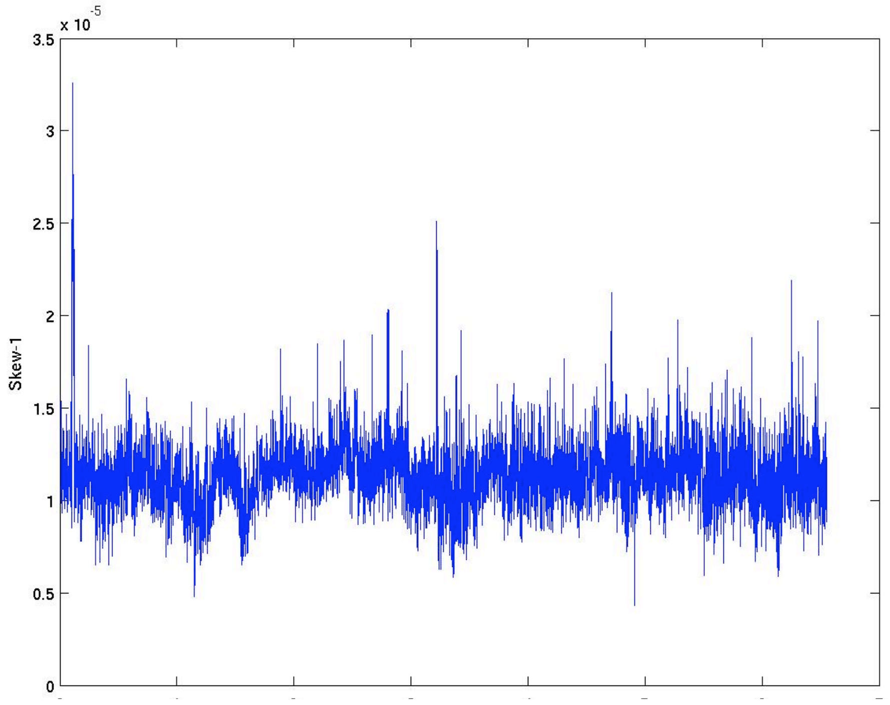
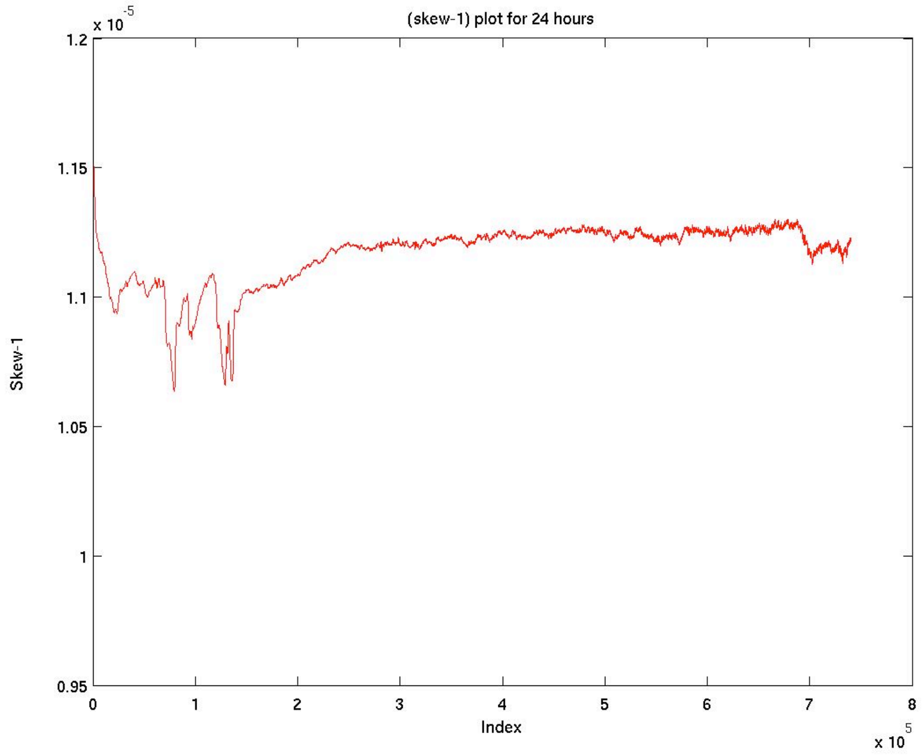


Fig 7: (β-1) plot for Zcomax Access Points for a period of 2 hours

We also calculated the (skew-1) for Zcomax access points for a period of two hours (Fig 7) and observed that the variability in the skew was of the order of 30 ppm which was a lot more than that obtained from the Cisco access points.

To further analyze how skew varies over large periods of time we took 24 hour readings of the beacon timestamps. The variability in the (skew-1) for Cisco Access Points was around 10 ppm for a period of 24 hours.



The conclusions which we could draw from the above experiment were

- β (skew) varies with time and is not constant.
- One of the factors contributing to the variability is temperature which is dependent on the load on the machine.

Estimation of β(skew)

From the above experiments we concluded that skew was not constant and varied with time and the variations were different for different hardware. So if we could correctly estimate the relative skew of the clocks then we can correctly estimate propagation delay between the two points. The basic equation that we used was

$$\tau_A'(t) - \tau_B'(t) = (\beta - 1)\,\tau_B'(t) + \varepsilon$$

- $\tau_A'(t) - \tau_B'(t)$ gives us the measured time difference between the send and the receive times
- $(\beta - 1)\,\tau_B'(t)$ gives the value of the estimated time difference using the estimated β
- ε is the error in the measured with respect to the calculated.

Below we explain the techniques which we used for the estimation of the β.

- **Initial Regression Slope:**
  In this method, we calculated β by taking the linear regression on first 300 readings ~ 30 seconds and used this value to estimate skew for the next 10 minutes and 2 hours.
- **Moving Average Beta:**

We used previous 20 readings to calculate the current b value and moved the window by 1 reading each time. Moving average method acts as a low pass filter used in signal processing and removes the effect of the short-term variation in skew and gave a good estimate of the present value of the skew.

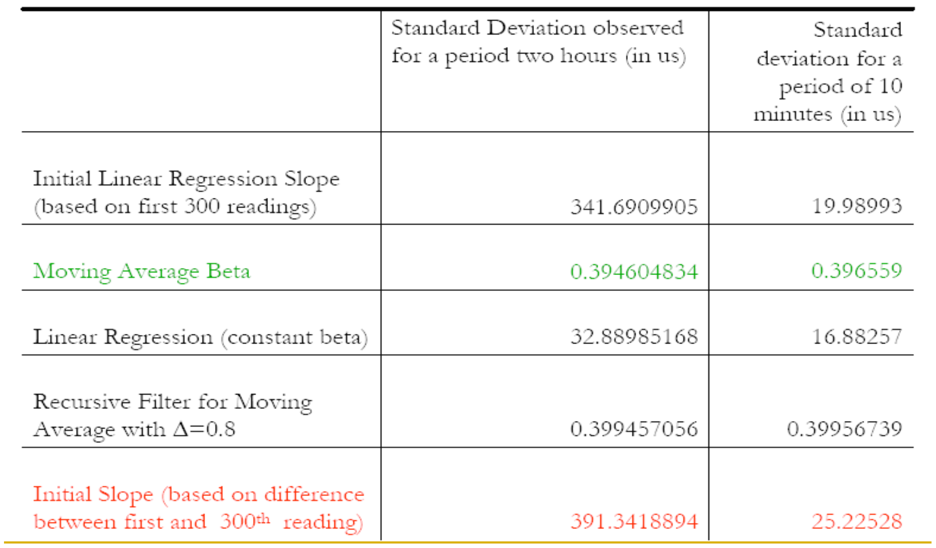
- **Linear Regression :**
  In this method, we calculated β by taking the linear regression of the all the past timestamp readings.

- **Recursive Filter for moving Average :**
  We tried to increase the performance of the moving average by taking the average of the past and the recent readings with a with Δ = 0.8,0.9.We didn't observe a significant change by varying Δ. The formula that we used was
  $$\beta_{new} = \beta_{old}(1-\Delta)+ \beta_{current}\Delta$$
- **Initial Slope based on the 1st and 300th reading:**

One of the other methods that we used were to take the slope of the send and the receive time differences of the 1st and the 300th readings and get an estimate of β.

Experimental Results

The experimental results which we obtained for the Cisco as well as the Zcomax Access Points are shown in tabular form below:



	Standard Deviation observed for a period two hours (in us)	Standard deviation for a period of 10 minutes (in us)
Initial Linear Regression Slope (based on first 300 readings)	341.6909905	19.98993
Moving Average Beta	0.394604834	0.396559
Linear Regression (constant beta)	32.88985168	16.88257
Recursive Filter for Moving Average with Δ=0.8	0.399457056	0.39956739
Initial Slope (based on difference between first and 300th reading)	391.3418894	25.22528

Table 3: Standard Deviation obtained from Cisco Access Point Readings

	Standard Deviation observed for a period two hours (in us)	Standard deviation for a period of 10 minutes (in us)
Initial Linear Regression Slope (based on first 300 readings)	672.1298	114.7298
Moving Average Beta	2.325077	2.658543
Linear Regression Constant Beta	215.884741	80.3820391
Recursive Filter for Moving Average with Δ=0.8	2.325087887	2.658738684
Initial Slope (based on difference between First and 300th reading)	686.9189	115.8336

Table 4: Standard Deviation obtained from Zicomax Access Point Readings

The results show that the Cisco Access Points have a more stable clock as compared to the Zicomax access points. It also proved that the Moving Average Beta method gave us the least variance in the noise (ε) and the initial slope method (based on the 1st and the 300th readings) gave the maximum deviation.

Conclusions

The conclusions which we could draw from our experiments were variability in the skew depends on the sender and the receiver hardware and the best way to estimate this variability is to use the moving average model, which as stated above removed the short term variability in the skew.

Future Work

Clock synchronization is one of the major issues faced in any real-time and distributed environment type of system. If we could accurately estimate the clock characteristics (mainly the skew) then we can synchronize any system of nodes. The novelty in our scheme is that we do not use any expensive clocks for synchronization. The future work includes increasing the accuracy of the pinpoint algorithm by using the moving average skew model. The final goal of the experiment is to make a more accurate and adhoc protocol for clock synchronization and replace the NTP (Network time Protocol).

References:
[1] M. Youssef, A. Youssef ,C.Rieger,U Shankar and A. Agrawala Pinpoint: An asynchronous Time-Based Location Determination System. *Proceedings of the 4th international conference on Mobile systems, applications and services.* pages 165-176,2006.
[2] P. Enge and P. Misra. Special issue on GPS: The Global Positioning System. *Proceedings of the IEEE*, 87(1):3–172, January 1999.
[3] V. N. Padmanabhan and L. Subramanian. Determining the Geographic Location of Internet Hosts. In *Proceedings of ACM SIGMETRICS*, pages 324–325, 2001.
[4] J. Werb and C. Lanzl. Designing a Positioning System for Finding Things and People Indoors. *IEEE Spectrum*, 35(9):71–78, September 1998.
[5] M. Youssef and A. Agrawala. The Horus WLAN Location Determination System. In *Third International Conference on Mobile Systems, Applications, and Services (MobiSys 2005)*, pages
205–218, June 2005.
[6] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS) Using AoA. In *Proceedings of IEEE Infocom*, volume 3, pages 1734– 1743, 2003.
[7]www.cacetech.com
[8]www.wireshark.org