NTP Clock Discipline Principles

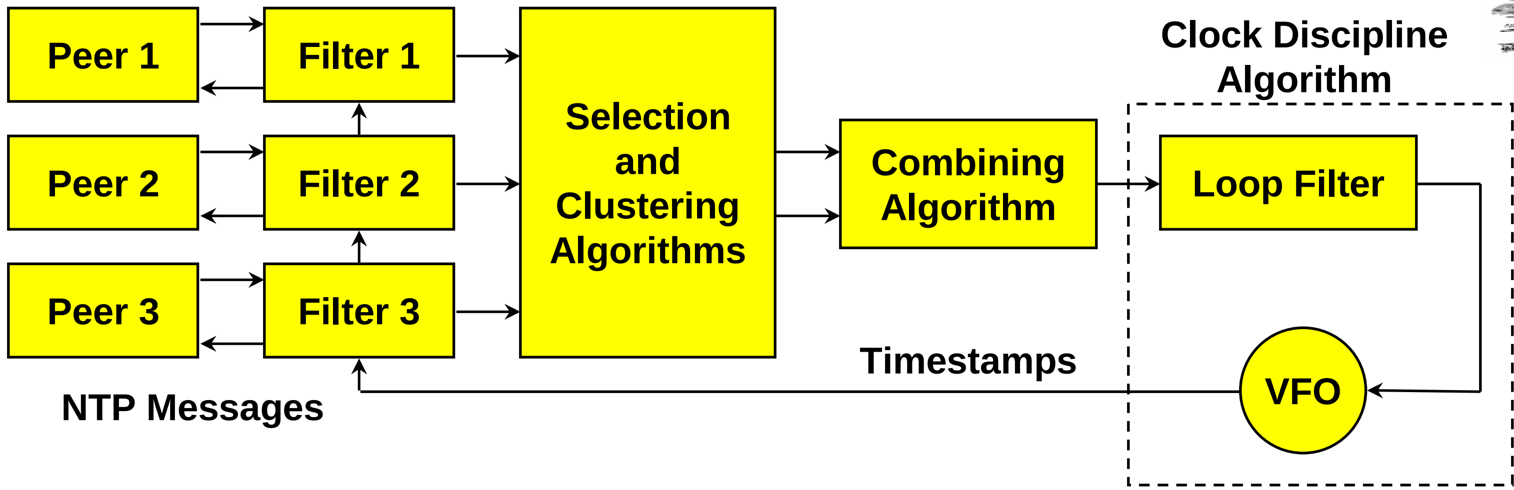
David L. Mills
University of Delaware
http://www.eecis.udel.edu/~mills
mailto:mills@udel.edu

Sir John Tenniel; *Alice's Adventures in Wonderland,*Lewis Carroll

Introduction

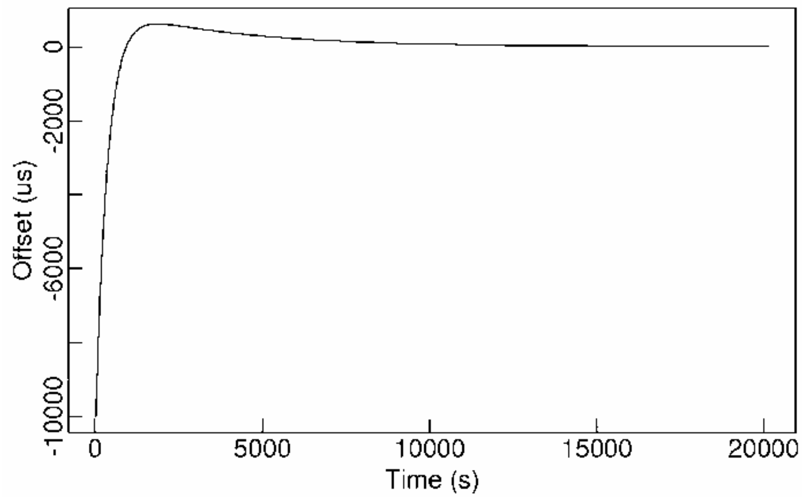
- Network Time Protocol (NTP) synchronizes clocks of hosts and routers in the Internet
- Well over 100,000 NTP peers deployed in the Internet and its tributaries all over the world
- Provides nominal accuracies of low tens of milliseconds on WANs, submilliseconds on LANs, and submicroseconds using a precision time source such as a cesium oscillator or GPS receiver
- Unix NTP daemon ported to almost every workstation and server platform available today - from PCs to Crays - Unix, Windows, VMS and embedded systems
- The NTP architecture, protocol and algorithms have been evolved over the last twenty years to the latest NTP Version 4 described in this briefing

NTP architecture overview

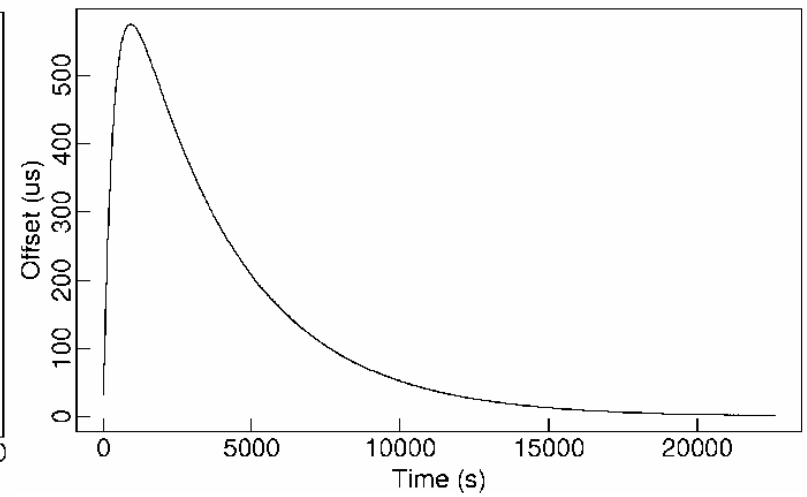


- Multiple servers/peers provide redundancy and diversity
- Clock filters select best from a window of eight time offset samples
- Intersection and clustering algorithms pick best *truechimers* and discard *falsetickers*
- Combining algorithm computes weighted average of time offsets
- Loop filter and variable frequency oscillator (VFO) implement hybrid phase/frequency-lock (P/F) feedback loop to minimize jitter and wander

Traditional approach using phase-lock loop (PLL)



Response to 10-ms Phase Step



Response to 2-PPM Frequency Step

- Left graph shows the impulse response for a 10-ms time step and 64-s poll interval using a traditional linear PLL.
- Right graph shows the impulse response for a 5-PPM frequency step and 64-s poll interval.
- It takes too long to converge the loop using linear systems.
- A hybrid linear/nonlinear approach may do much better.

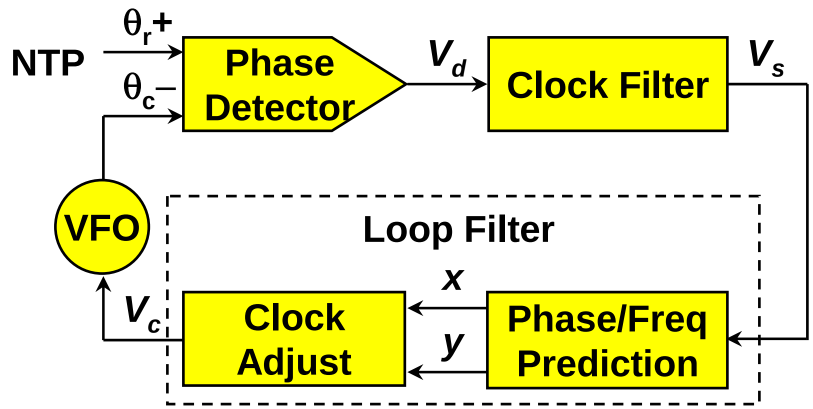
Clock discipline design principles

- The clock discipline algorithm functions as a nonlinear, hybrid phase/frequency-lock (NHPFL) feedback loop.
- Detailed computer clock analysis yields the optimum averaging interval depending on prevailing network jitter and oscillator wander.
- Optimum value is determined in real time by measuring the jitter and wander separately.
- Clock state machine quickly converges time and frequency and suppresses transients resulting from leap events, etc.
- Huff&puff algorithm corrects for large outlyers and asymmetric delays
- Popcorn spike suppressor clips noise spikes.

Clock discipline design approach

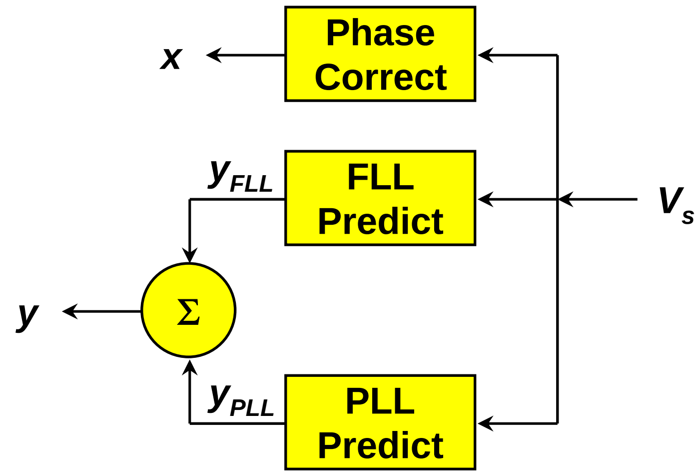
- Phase noise due to network jitter prevails at the lower poll intervals, so a second-order phase-lock loop (PLL) is the best frequency predictor.
- Frequency noise due to random-walk oscillator wander prevails at the higher poll intervals, so a first-order frequency-lock loop (FLL) is the best frequency predictor.
- A crafted heuristic algorithm is necessary to combine both predictions.
- The NHPFL algorithm combines the time and frequency predictions in a seamless way for poll intervals from 16 seconds to 36 hours.
  - The PLL frequency adjustment is computed as the integral of past frequency offsets.
  - The FLL frequency adjustment is computed as the exponential average of past frequency offsets.
  - An additional phase adjustment is necessary for loop stabiility.
  - The poll interval, which determines the loop time constant, is determined in response to measured jitter and wander.

Clock discipline algorithm



- V_d is a function of the NTP and VFO phase differences
- V_s depends on the stage chosen of the clock filter shift register
- x is the phase correction and y the frequency adjustment computed by the prediction functions
- The clock adjust process runs once each second to adjust the VFO phase by V_c
- The loop behavior is determined by the loop filter parameters

FLL/PLL prediction functions



- V_s is the phase offset produced by the clock filter algorithm
- x is the phase correction computed as the value of V_s
- y_{FLL} is the frequency prediction computed as the average of past values of V_s
- y_{PLL} is the frequency prediction computed as the integral of past values of V_s
- y_{FLL} and y_{PLL} are combined according to weight factors determined by poll interval, update interval and Allan intercept

Detailed calculations

- The phase correction x and frequency predictions y_{PLL} and y_{FLL} are recalculated at each clock update.

$$y_{PLL} = \frac{\min(\mu, 2^{\tau})\theta}{(4K_{PLL}2^{\tau})^2} \qquad y_{FLL} = \frac{\Delta_{\theta}}{K_{FLL}\max(\mu, A_x)} \qquad x = \theta \qquad \begin{array}{ll} \zeta = 1.2 & 2^{\tau+4} > A_X \\ \zeta = 1 & \text{otherwise} \end{array}$$

- The VFO adjustment V_C is updated by the clock adjust process at one-second intervals.

$$dx = \frac{x}{K_{PLL}2^{\zeta\tau}} \qquad V_C = y_{PLL} + y_{FLL} + dx \qquad x = x - dx$$

- Constants

K_{PLL} = 16	PLL gain
K_{FLL} = 8	FLL gain
A_x = 1024s	Allan intercept

- Variables

τ	poll interval ($\log_2$)
μ	update interval
θ	clock offset
Δ_{θ}	offset change since last update
ζ	damping factor

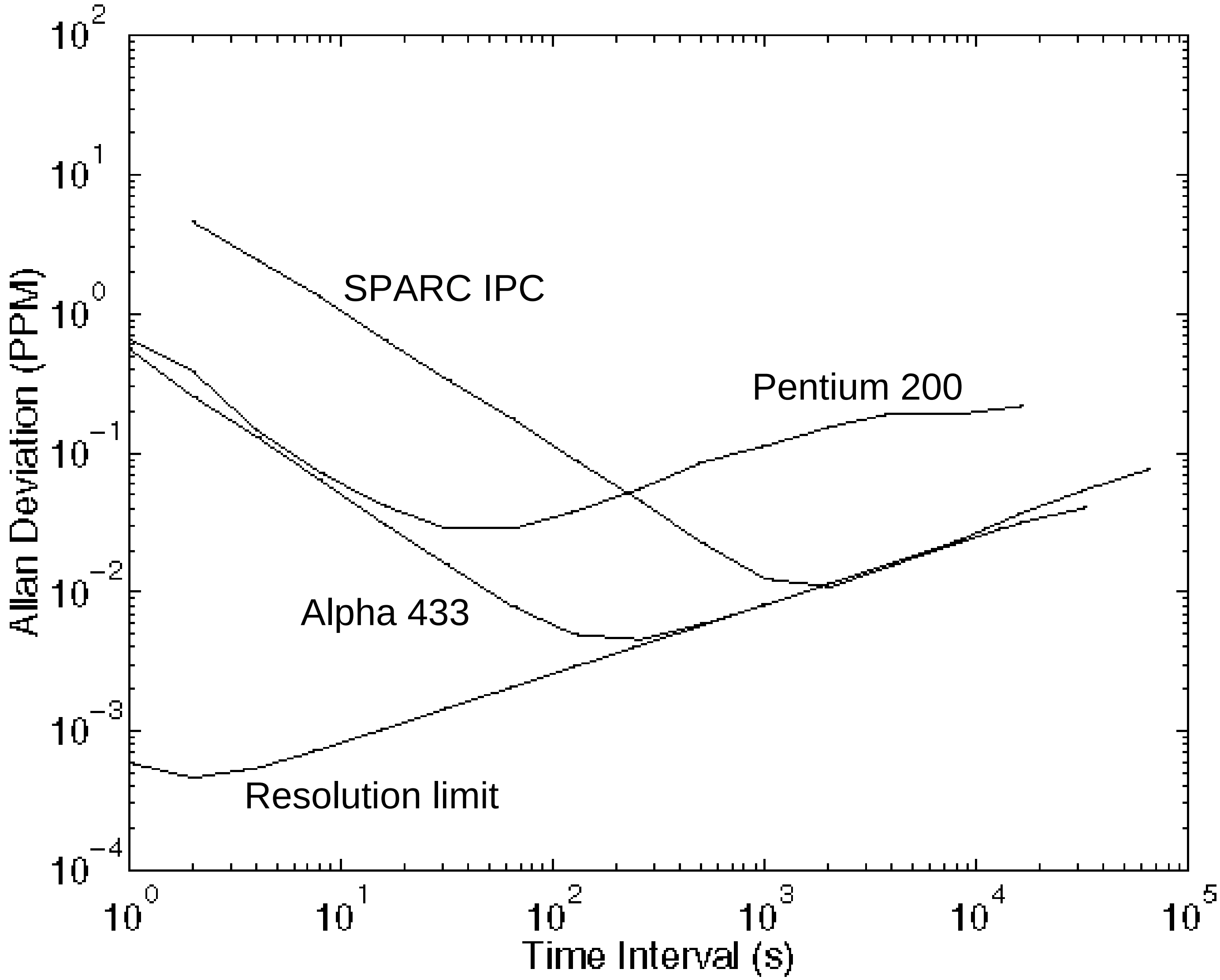
Network clock discipline error analysis

- Network jitter is a random process with parameter ρ.
  - The ρ is the RMS jitter measured by the clock filter algorithm.
  - Jitter process is closely mimicked by an exponential distribution with parameter ρ.
  - Jitter characteristic is a straight line with slope –1 in log-log coordinates.
- Oscillator wander is the integral of a random process with parameter σ.
  - The σ is the exponential average of past frequency adjustments.
  - Wander process is closely mimicked by the integral of a zero-mean normal distribution with standard deviation σ.
  - Wander characteristic is a straight line with slope +0.5 in log–log coordinates.
- Allan intercept is defined as the intersection of the jitter and wander characteristics.
- The intersection coordinates define the optimum averaging interval and poll interval.

Constructing Allan deviation characteristics

- Time differences between the system clock and an external standard are measured at 1-s intervals over several days
- For a given time interval τ the frequency $y(\tau)$ is determined as the time difference between the beginning and end of the interval divided by τ
- The Allan deviation $\sigma_y(\tau)$ is defined as the average of successive frequency differences $\Delta y(\tau)$ as τ varies from 1 s to several days.
- The Allen deviation plot appears in log-log coordinates as two intersecting lines determined by the jitter and wander characteristics
- The following graph shows $\sigma_y(\tau)$ for three architectures and operating system, plus a synthesized characteristic with nanosecond resolution and assumed “good” frequency stability.
  - Alpha 433 has nanokernel modifications and 2.3-ns resolution.
  - Pentium 200 has nanokernel modifications and 5-ns resolution.
  - SPARC IPC has microkernel modifications and 1000-ns resolution.

Allan deviation characteristics compared



Allan intercepts compared

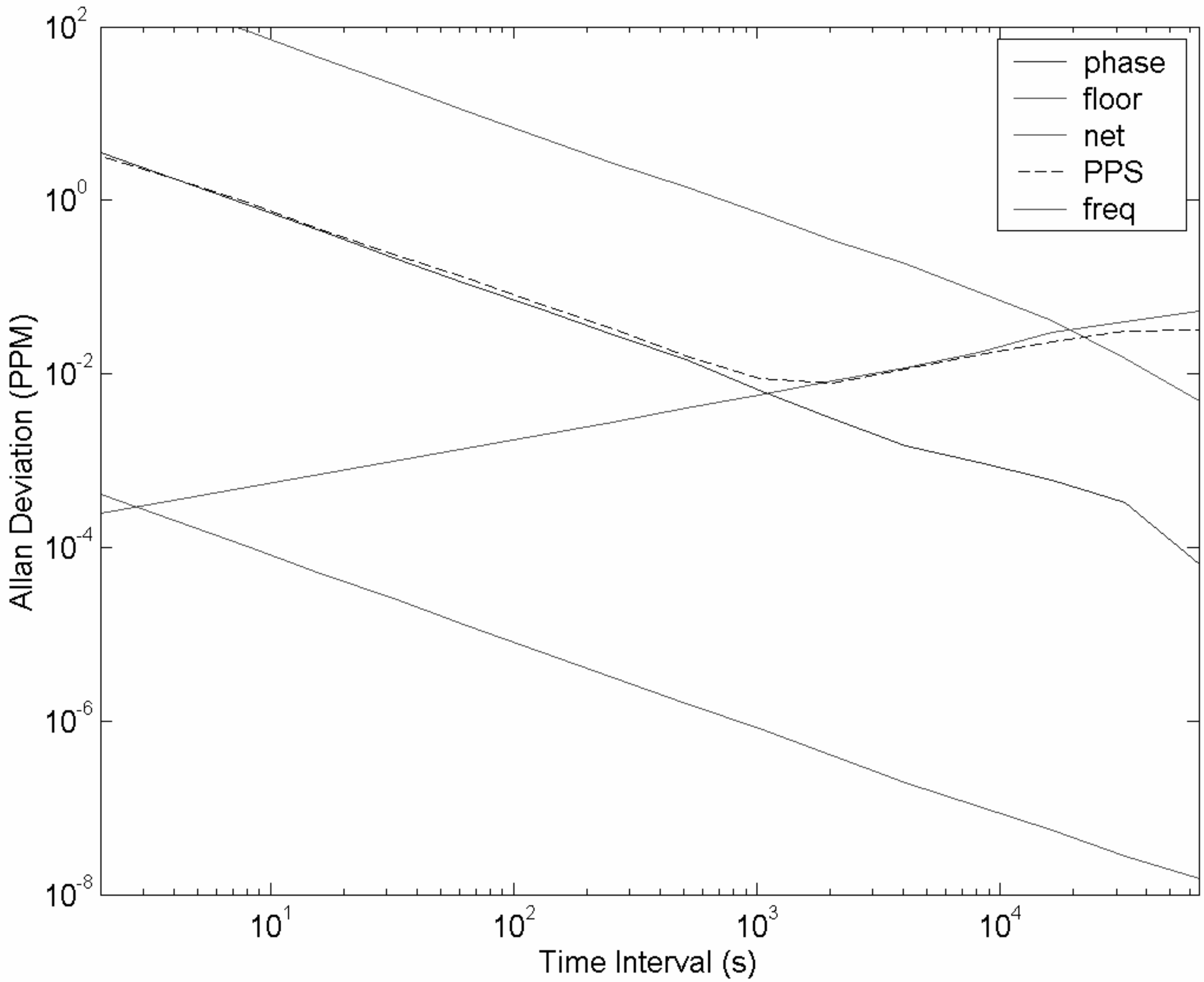
System	Resolution	Precision	Stability	*x* Intercept	*y* Intercept	Range *
SPARC IPC	1000 ns	1000 ns	good	2000 s	.01 PPM	600 - 5000 s
Pentium 200	1 ns	5 ns	poor	50 s	.03 PPM	10 - 300 s
Alpha 433	1 ns	2.3 ns	good	200 s	.005 PPM	50 - 2000 s
Resolution limit	1 ns	1 ns	good	2 s	.0004 PPM	1 - 10 s

* For stability no worse than twice *y* intercept

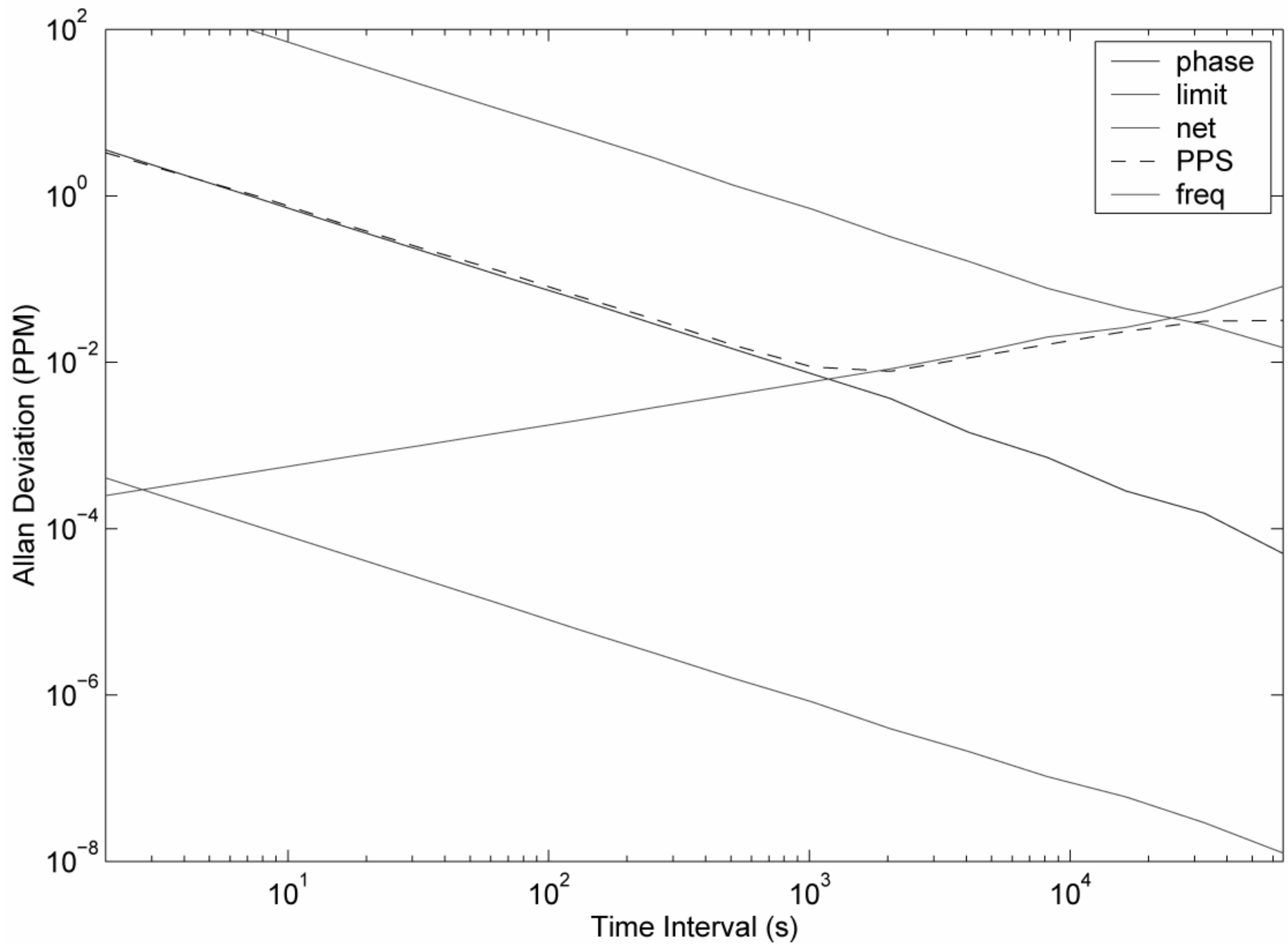
Allan deviation cont.

- A useful performance predictor can be constructed from Allan deviation plots and synthetic noise sources. The graph on the next page compares the Allan deviation of a PPS source to pseudo-random noise sources.
  - The PPS signal is connected to a Sun SPARC IPC running SunOS 4.1.3.
  - Trace *PPS* shows the measured combined phase (slope -1) and frequency (slope +0.5) noise.
  - Trace *net* is generated from an exponential distribution with parameter 500e-6. This is typical of a workstation synchronized to a primary time server over the Internet.
  - Trace *phase* is generated from an exponential distribution with parameter 5e-6. Note how closely this matches the *PPS* phase characteristic.
  - Trace *floor* is generated from a uniform distribution between 0 and 2 ns. This may represent the best achievable with modern workstations.
  - Trace *freq* is generated from the integral of a zero-mean normal distribution with parameter 5e-10. This represents the random-walk characteristic of typical computer oscillators.

Allan deviation calibration



test


phase
limit
net
PPS
freq
10^2
10^0
10^-2
10^-4
10^-6
10^-8
10^1
10^2
10^3
10^4
Allan Deviation (PPM)
Time Interval (s)

Poll adjust strategy

- Note that as τ increases the phase noise $\langle\varphi_P\rangle$ decreases with slope −1, while the frequency noise $\langle\varphi_F\rangle(\tau << 1)$ increases with slope +0.5. Thus, the minimum error is when $\langle\varphi_P\rangle = \langle\varphi_F\rangle(\tau << 1)$. (Remember that τ is $\log_2$ of the actual poll interval.) Thus, the strategy is:
- If $\langle\varphi_P\rangle > \langle\varphi_F\rangle(\tau << 1)$ and $|\theta| < K_G\langle\varphi_P\rangle$, increase the hysteresis counter h by τ.
  - If $h > K_H$, set $h = 0$ and increase τ by one.
- Else, decrease h by two, in order to adapt to rapid frequency changes.
  - if $h < -K_H$, set $h = 0$ and decrease τ by one.

- Constants

$K_H = 30$	hysteresis limit
$K_G = 4$	hysteresis threshold

- Variables

$\langle\varphi_P\rangle$	average phase differences
$\langle\varphi_F\rangle$	average frequency differences
h	hysteresis counter

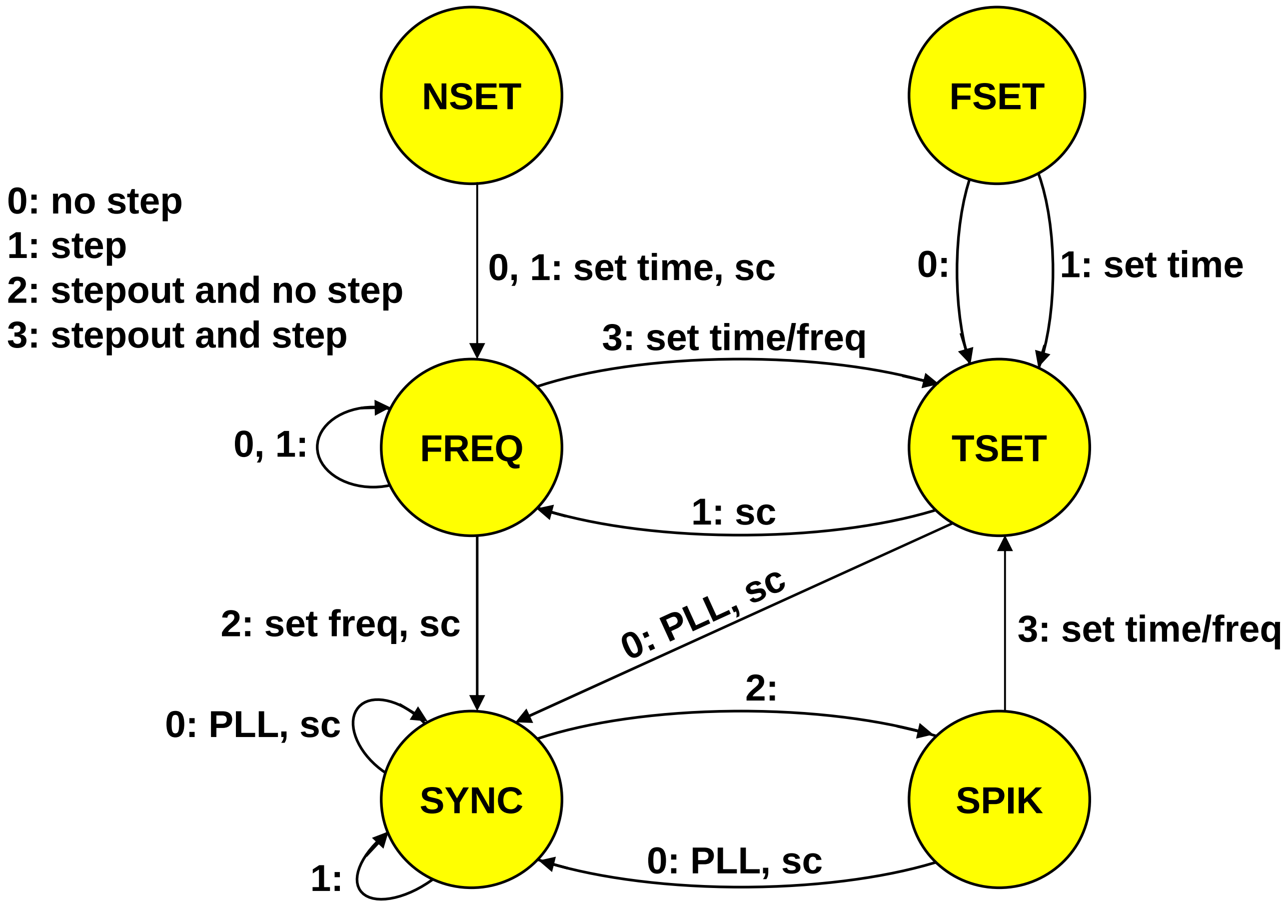
State machine operations

- There are three thresholds which affect the state machine
  - Panic threshold (1000 s): exit to the operating system if offset exceeds
  - Step threshold (128 ms): ignore if offset exceeds until stepout
  - Stepout threshold (900 s): interval within which step spikes are ignored
- When the discipline is started for the first time, set the time and calculate a possibly large frequency correction.
- Subsequently when the discipline is started, set the time only if the offset exceeds the step threshold.
- When calculating the frequency correction, continue to the stepout threshold in order to produce an accurate value, then set the time and frequency.
- Once the initial time and frequency have been set, run the HNPFL algorithm and the poll-adjust algorithm. Ignore transients greater than the step threshold, unless the stepout threshold is exceeded.

Clock state machine

state	< step	>step <stepout	>step >stepout	comments
NSET	FREQ set time	FREQ set time	na	freq file does not exist
FSET	TSET	TSET set time	na	freq file exists
TSET	SYNC PLL/FLL	FREQ	na	set time
FREQ	SYNC set freq	FREQ	TSET set time/freq	set freq
SYNC	SYNC PLL/FLL	SYNC	SPIK	normal
SPIK	SYNC PLL/FLL	na	TSET set time/freq	popcorn suppressor

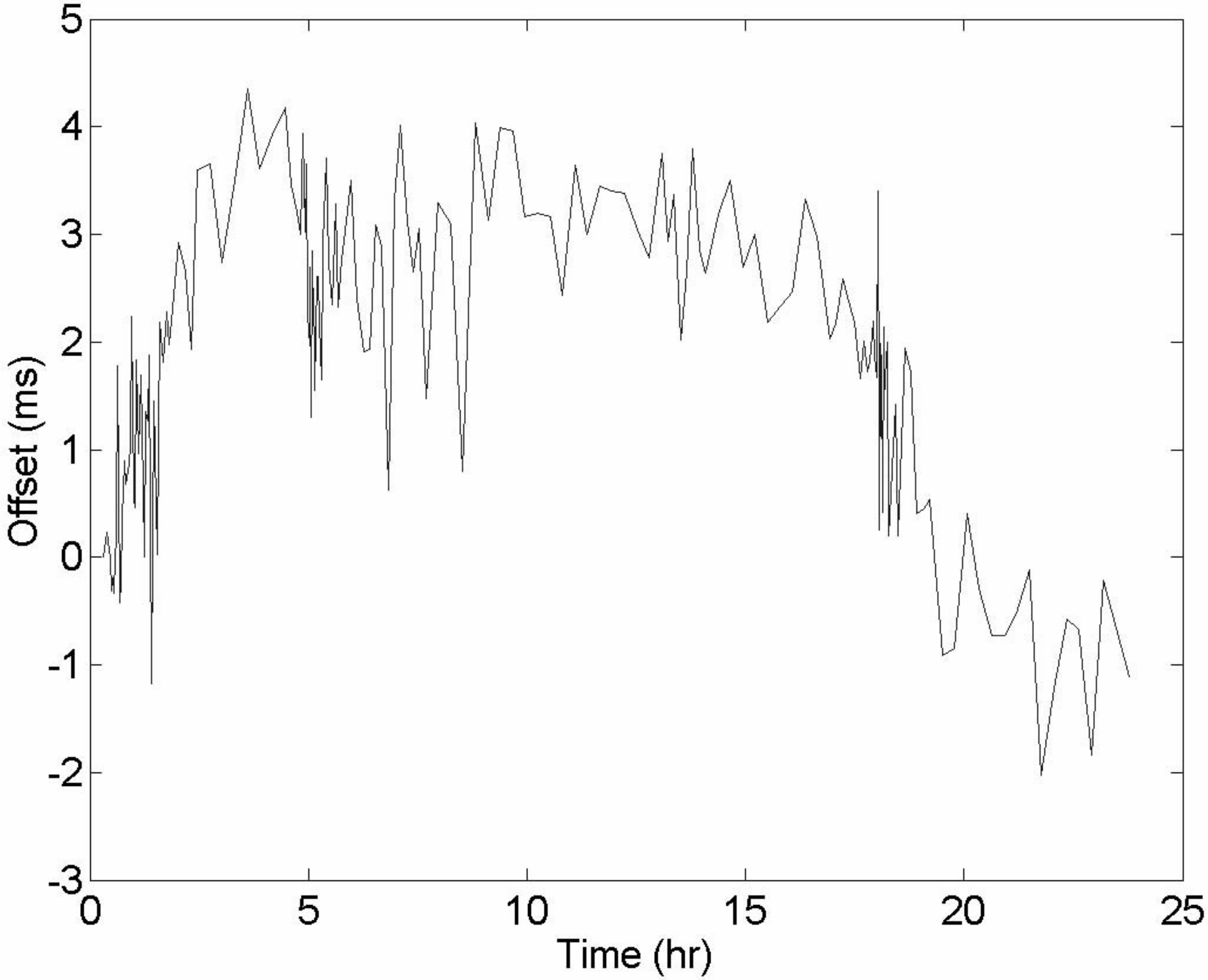
Clock state machine transition function



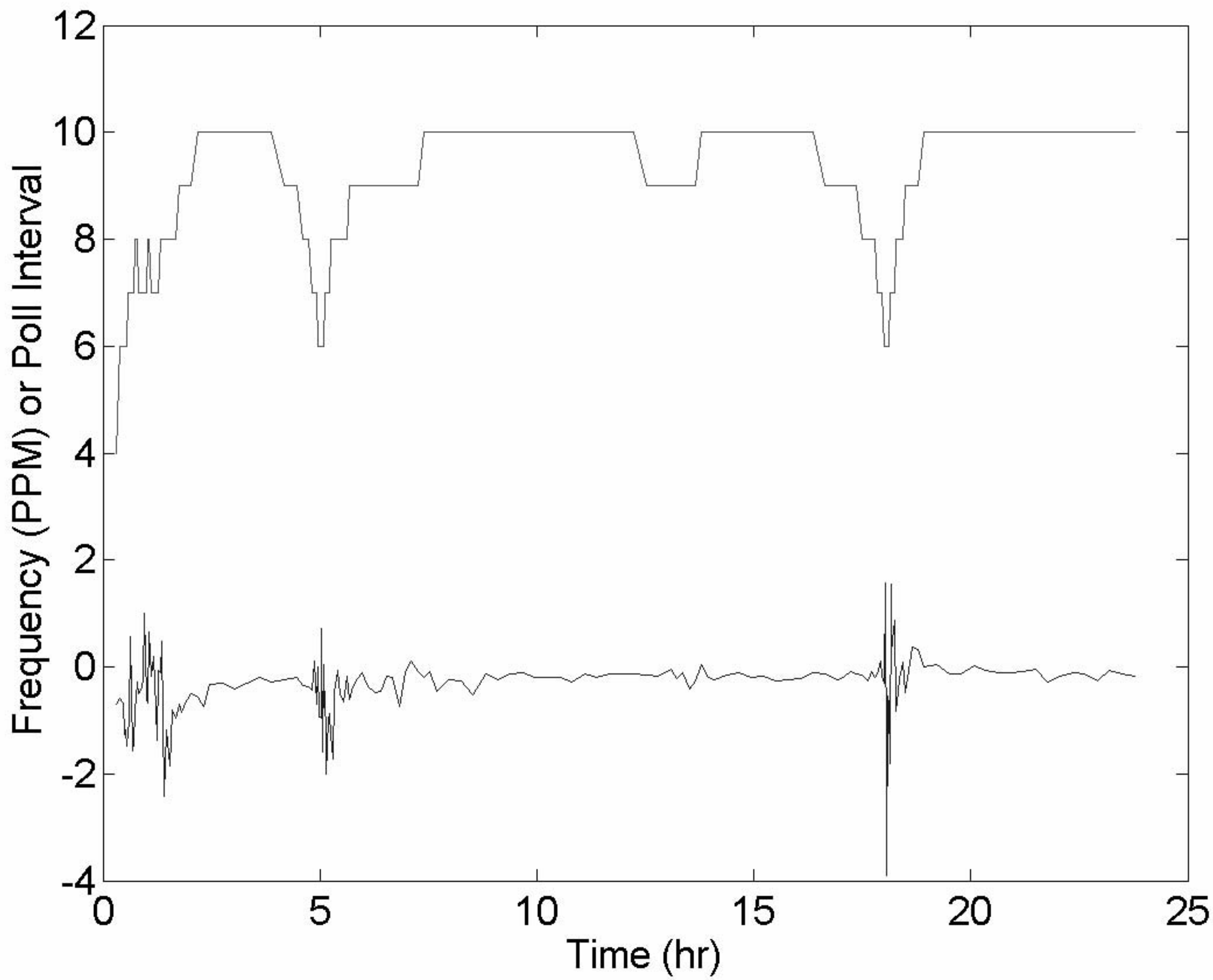
Clock discipline algorithm performance

- The algorithm converges time within 5 ms and frequency within 2 PPM in a very short time with poll intervals up to 10 (1024 s).
  - Time to converge with no frequency file is less than 20 min.
  - Time to converge with frequency file and no iburst is less than 4 min.
  - Time to converge with frequency file and iburst is less than 10 s.
  - Previous designs could take days to achieve this performance.
- Following slides show results from a simulator run for typical LAN
  - Initial oscillator frequency offset -400 PPM with wander parameter 1 s/s.
  - Initial time offset 600 s with network jitter parameter 1 ms.
  - These are parameters typical for 10 Mb Ethernets and computer oscillators.
- The poll interval rapidly adapts to frequency changes.
  - The frequency (blue) is in PPM.
  - The poll interval (green) is in log2(s) units.
  - It increases slowly it jitter is greater than wander and decreases rapidly otherwise.

Clock offset from simulator



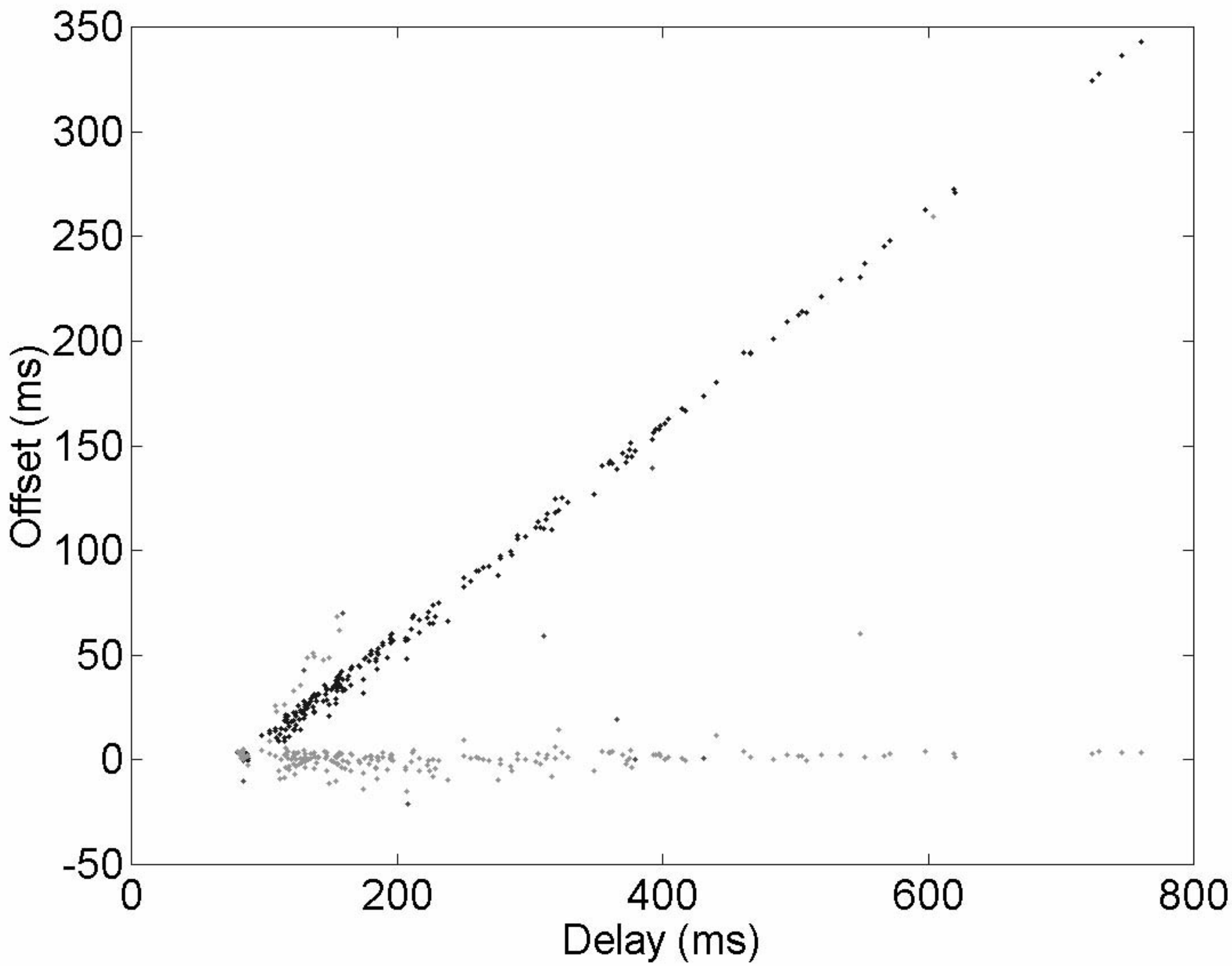
Frequency offset and poll interval from simulator



Huff&puff filter

- Many network paths show large delays and delay variations on one direction of transmission but not the other.
- These conditions often prevail only during some period of the workday.
- A wedge scattergram plotting offset versus roundtrip delay samples is shown in the next slide:
  - Blue dots represent the clock filter output.
  - Green dots represent the huff&puff filter output.
  - Red dots are discarded by the popcorn spike suppressor.
- Let (θ_0, δ_0) be the apex coordinate at the minimum roundtrip delay and (θ, δ) the coordinate of a blue dot on the positive limb. Then, (θ', δ), where $\theta' = \theta - (\delta - \delta_0) / 2$, is the coordinate of the corresponding green dot and θ' is the corrected offset produced by the huff&puff filter.
- A similar argument holds for the negative limb.

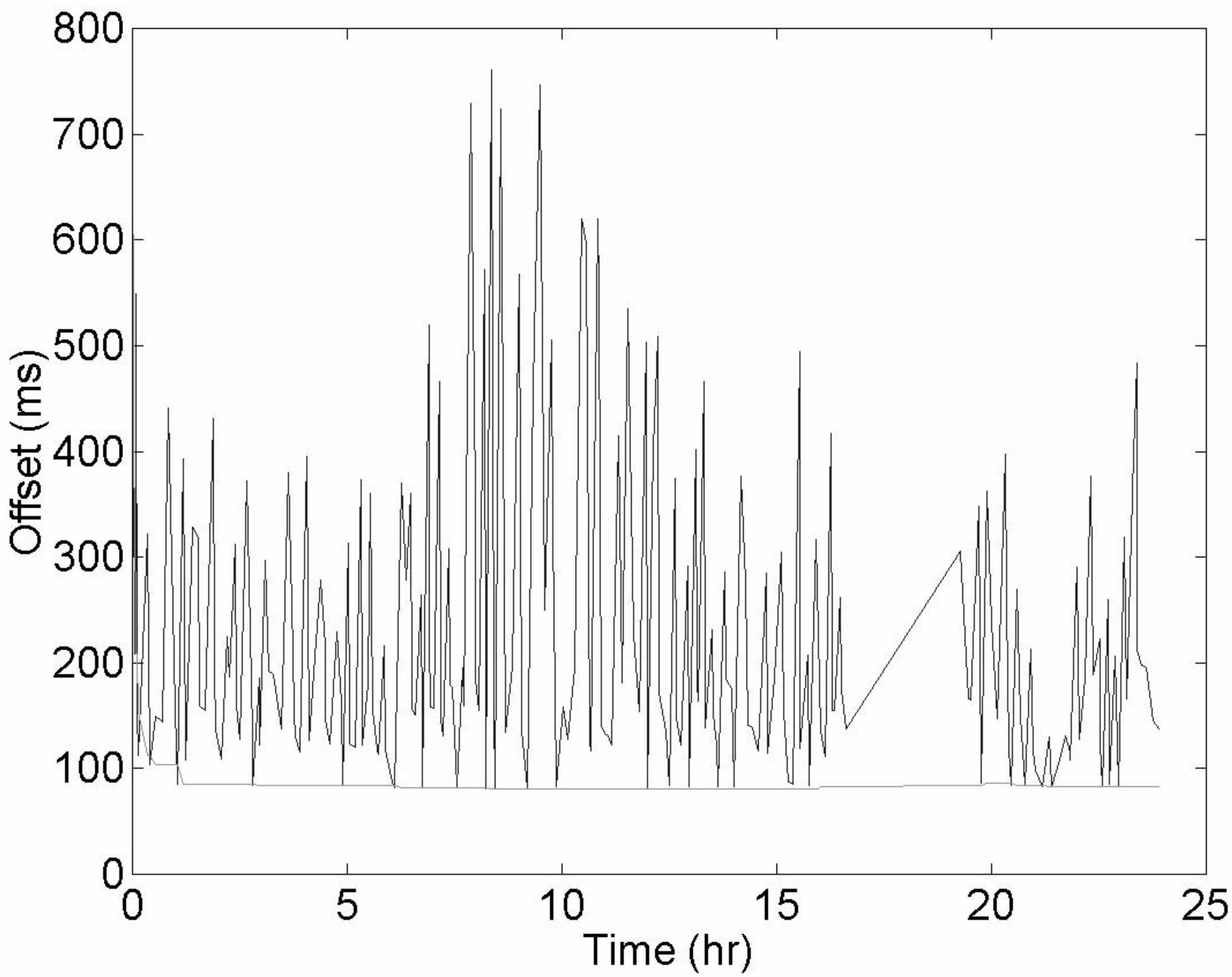
Huff&puff wedge scattergram



Huff&puff minimum delay estimator

- The time series graph shown on the following slide shows the sample delay (blue trace) together with the minimum delay over a window extending four hours in the past (green trace).
- This is typical behavior for a moderately loaded network link, whether or not asymmetrical delays are present.
- The server was apparently unreachable between hours 16-19.

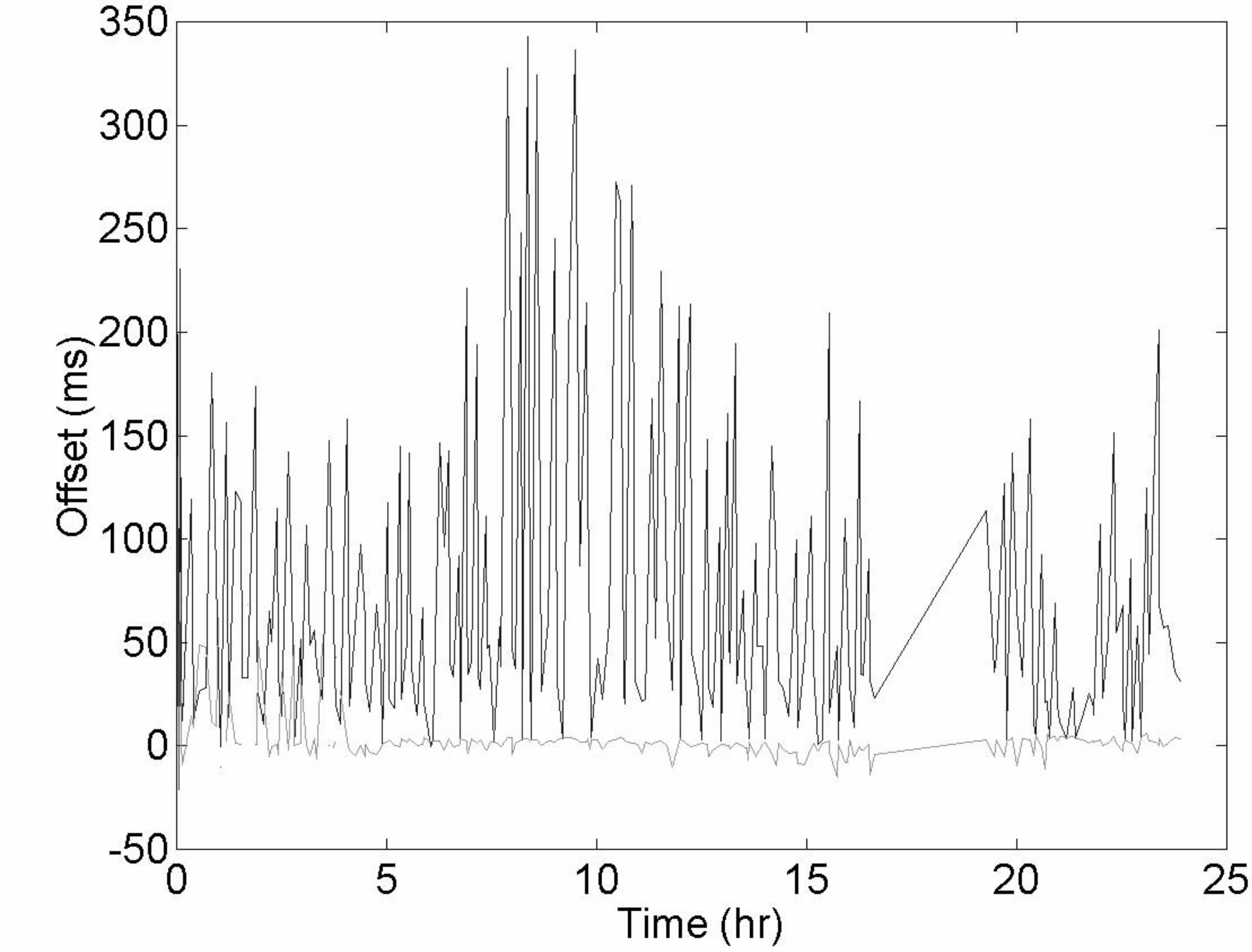
Huff&puff delay time series



Huff&puff filter performance

- The time series graph shown on the following slide shows the clock filter output (blue trace) and corresponding huff&puff filter output (green trace).
- The popcorn spike suppressor discards samples where the absolute sample-sample offset difference exceeds the running average of RMS jitter in the clock filter output.
- While this particular scenario shows a dramatic reduction in jitter and improvement in accuracy, other scenarios show less improvement, including:
  - The minimum delay statistic cannot be reliably determined if the most recent minimum delay sample is beyond the window.
  - The delays are large and more symmetric, so the sample point does not occur on a positive or negative limb.
  - The popcorn spike suppressor fails to detect and discard the outlyers.

Huff&puff offset time series



NTP online resources

- Network Time Protocol (NTP) Version 3 Specification RFC-1305
  - NTPv4 features documented in release notes and reports cited there
- Simple NTP (SNTP) Version 3 specification RFC-2030
  - Applicable to IPv4, IPv6 and ISO CNLS
- List of public NTP time servers (as of May 2001)
  - 107 active primary (stratum 1) servers
  - 136 active stratum 2 servers
- NTP Version 4 implementation and documentation for Unix, VMS and Windows
  - Ported to over two dozen architectures and operating systems
  - Utility programs for remote monitoring, control and performance evaluation
  - Complete documentation in HTML format
- Collaboration resources at http://www.eecis.udel.edu/~mills/resource.htm

Further information

- Network Time Protocol (NTP): http://www.ntp.org/
  - Current NTP Version 3 and 4 software and documentation
  - FAQ and links to other sources and interesting places
- David L. Mills: http://www.eecis.udel.edu/~mills
  - Papers, reports and memoranda in PostScript and PDF formats
  - Briefings in HTML, PostScript, PowerPoint and PDF formats
  - Collaboration resources hardware, software and documentation
  - Songs, photo galleries and after-dinner speech scripts
- FTP server ftp.udel.edu (`pub/ntp` directory)
  - Current NTP Version 3 and 4 software and documentation repository
  - Collaboration resources repository
- Related project descriptions and briefings
  - See “Current Research Project Descriptions and Briefings” at http://www.eecis.udel.edu/~mills/status.htm