
CHAPTER 4

A Mirage of NTP

This is an application of the Mirage model to an existing protocol, the Network Time Protocol (NTP). We chose this protocol for its simplicity and its prevalence. Here we apply the abstract principles of Mirage and perform an experiment that helps further refine the Mirage model.

This analysis demonstrates the equivalence between latency variability and imprecision in the local perception of the remote state, thus extending the domain of the Mirage model to variable latency regimes. We also examined how Mirage relies on time measurements, due to the seeming contradiction in modeling a clock protocol with a time-based model.

We concluded that several optional components of the NTP specification are integral to our model of NTP in Mirage, and that these components should therefore be required. These components include the logical clock, peer dispersion, and data filter algorithms.

The Mirage model of NTP reduces to a hardware clock signal where delay variability is zero. This implies that the variability of the latency limits clock synchronization, rather than the overall latency magnitude. This reduction resulted in a refinement of the variability of the measured offset.

NTP has proven useful in demonstrating the application of some basic components of Mirage. Unfortunately, many of the novel aspects of Mirage are not applicable to NTP, e.g., guarded messages, state space partitioning, and communicability tradeoffs. Later chapters of this dissertation (μ -Net in Chapter 5, and μ -Scope in Chapter 6) demonstrate those components that NTP cannot.

4.1. An overview of NTP

We begin with a summary of NTP. Whereas there have been 4 versions of NTP (numbered 0 through 3), this exercise focuses on core aspects of the protocol, which have changed little in recent versions. Thus Version 3 is referenced here [Mi90b].

The Network Time Protocol is a protocol for synchronizing system clocks. It consists of a request/response engine for exchanging timestamp messages, a logical clock providing an adjustable time reference, and a set of algorithms for integrating sets of message responses to determine an appropriate adjustment.

The logical clock provides a mechanism for a user-adjustable time reference, if not already explicit in the node's operating system. The NTP logical clock engine is named the *Fuzzball*. The clock consists of an absolute offset, called an *epoch*, and a frequency ratio scale. The logical clock is defined as the epoch plus the hardware elapsed time multiplied by the frequency ratio, so an otherwise fixed system clock can be adjusted in both absolute offset and frequency. *Drift*, the variability in frequency, is not compensated for in the logical clock provided in NTP, and is assumed to be zero. Furthermore, the logical clock is described as 'optional' in the NTP specification. Between two logical clocks, *offset* is the difference in epochs, *skew* is the difference in frequency. *Dispersion* is the known error in the local clock.

NTP also includes a mechanism for organizing a set of clock servers into a hierarchy based on clock *strata*, where stratum is a group of clocks at a specified precision. Descriptions of various strata are given in [Mi90b]. Lower strata (e.g., Stratum 1) describe more accurate clocks; strata range from 1 to 255, with 0 representing an unspecified stratum. Stratum 1 clocks are called primary servers; strata 2-255 are called secondary. Stratum 1 clocks are the root(s) of the timeserver hierarchy, and a stratum k server is defined as having a distance of $k - 1$ to the nearest root.

The core of the NTP protocol is a client/server engine that stamps messages when sent and received. A message initiated at a host returns with 3 timestamps in its packet body (sender out ‘originate’, receiver in ‘receive’, receiver out ‘transmit’) and a fourth stamp retained upon receipt of the message, but separate from it (sender in ‘input’) (Figure 4.1). Received messages are automatically replied, and sent messages are periodically initiated according to local state information.

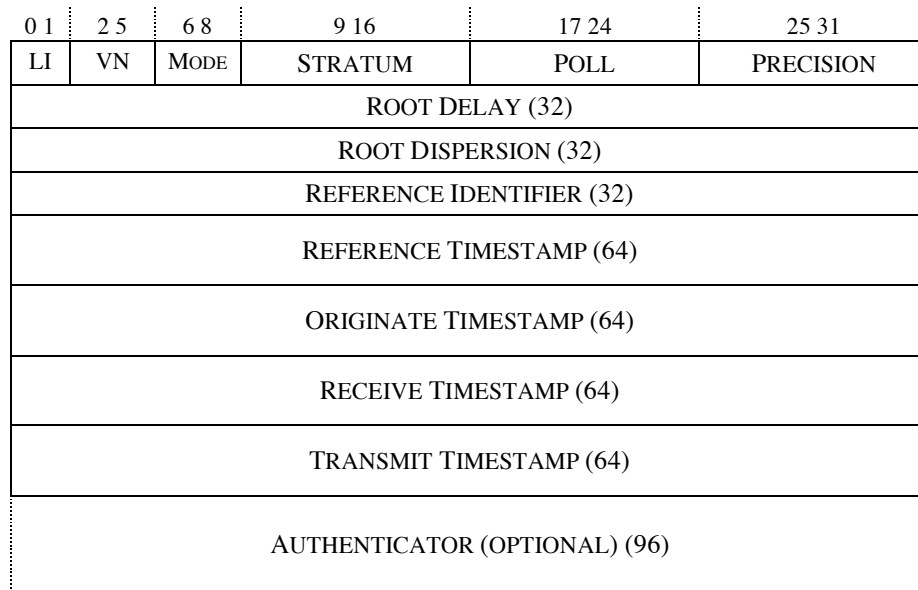


FIGURE 4.1
NTP message format

The local state information is combined with sets of replies. These replies are processed to prevent individual server clock errors from affecting clocks of their clients, where possible. Various combining and filtering algorithms provide this processing, and adjust the current local clock and maintain the clock server hierarchy. The hierarchy determines which servers will be consulted for future clock data.

There are two types of sets of message replies considered in NTP. Temporal sets of data are taken at different times from the same server; ensemble sets are taken at (nearly) the same time from a set of different servers.¹ Temporal sets are held in a shift register,

¹The terms ‘temporal set’ and ‘ensemble set’ occur in both statistics and physics, as they are used here.

and combined using a *filter algorithm*; ensemble sets are combined using the *peer selection algorithm* and the *combining algorithm*.

Other components of NTP include control messaging, authentication, and asymmetric modes of operation. The conventional symmetric mode of operation is outlined above; asymmetric modes include servers broadcasting to a set of workstation clients, and those clients, as well as root servers (read-only clocks). Control messaging provides user-level access to protocol state, for remote monitoring and manipulation. Authentication provides security and additional protection from Byzantine failures [Pe80]. These components are optional in the NTP specification. They are not considered in this analysis because they are supplemental to the basic operation of the protocol.

4.1.1. How NTP reads a clock

NTP reads a remote clock by sending a NTP message to the remote node (Figure 4.1), and waiting for a reply. Upon receipt of the reply message, the client/server engine pairs the incoming packet with the current local time ('input time', a.k.a. 'peer receive'), and processes it. The message is stamped at each reception and emission (Figure 4.2).

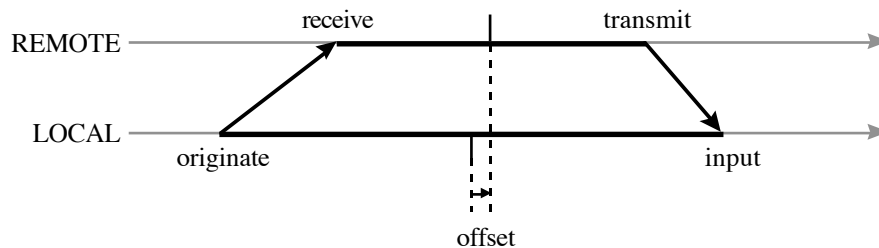


FIGURE 4.2
NTP message exchange

The four timestamps – originate, receive, transmit, and peer receive (input) – are used to compute round trip time and offset measurements. The round trip time (δ) can be computed from the difference between the sender interval and the receiver interval (Equation 4.1). This formula appears in the NTP specification.

'Clock time' is defined as the average over an interval, i.e., local time is the average of the originate and input values, and remote time is the average of the receive and transmit values (Equations 4.2, 4.3). 'Offset' (θ) is defined as the difference between the remote clock time and the local clock time, where a positive offset indicates that the

remote clock is ahead of the local clock (Equation 4.4, Figure 4.2). This formula is derived differently than in the NTP specification, but the result is the same (Equations 4.5, 4.6).

$$\text{Equation 4.1: } \delta = (T_{in} - T_{orig}) - (T_{xmit} - T_{recv})$$

$$\text{Equation 4.2: } T_{local} = \frac{(T_{in} + T_{orig})}{2}$$

$$\text{Equation 4.3: } T_{remote} = \frac{(T_{xmit} + T_{recv})}{2}$$

$$\text{Equation 4.4: } \theta = T_{remote} - T_{local}$$

$$\text{Equation 4.5: } \theta = \frac{(T_{xmit} + T_{recv})}{2} - \frac{(T_{in} + T_{orig})}{2}$$

$$\text{Equation 4.6: } \theta = \frac{(T_{recv} - T_{orig}) + (T_{xmit} - T_{in})}{2}$$

The calculated round trip delay is ‘exact’, i.e., the four timestamps exactly determine the total round trip time. The offset calculation, however is not similarly exact. The NTP formula for offset makes several assumptions: (1) that clock time varies only linearly during the exchange, and (2) that outgoing transit time and incoming transit times are identical. Assumption (1) is required – were it not, the protocol would not be able to determine a clock value. The clock value is thus defined to be the linear interpolation of the send and receive times of the message.

The calculated offset is affected by the difference between the outgoing and incoming transit times. Outgoing transit time adds to the measured offset, whereas incoming transit time subtracts from the measured offset. Under no other assumptions, the one-way delays are bounded between 0 and the total round trip time (δ) (Equation 4.7). This loose bound on delay results in a similarly loose constraint on calculated offsets (Equation 4.8). These bounds are shown visually (Figures 4.3, 4.4). Figure 4.3 shows that if outgoing delay is larger than incoming delay, the offset is incorrectly

measured as larger than intended; if incoming delay dominates the round trip time (Figure 4.4), the offset is incorrectly measured as smaller (falsely negative).

Other boundaries exist in the specification that require local time lines to be non-negative, a receive time between the originate and transmit times, and a transmit time between the receive and input times as well.

Equation 4.7: $0 \geq \delta_{unidirectional} \geq \delta_{bidirectional}$

Equation 4.8: $\theta + \frac{\delta}{2} \geq \hat{\theta} \geq \theta - \frac{\delta}{2}$

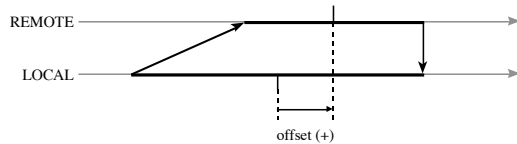


FIGURE 4.3
Exchange slid forward (maximum offset)

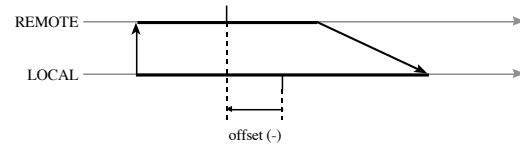


FIGURE 4.4
Exchange slid backward (minimum offset)

4.1.2. NTP background

NTP relies on the UDP protocol for connectionless transport-layer services [Po80], i.e., unreliable datagram services. UDP provides user-level access to the IP protocol [Po81b] and is the basis of the request/response transport mechanism, but the NTP protocol does not require either IP or UDP specifically. NTP evolved from earlier time protocol extensions of the Internet protocol suite, specifically the Time Protocol [Po83] and ICMP Timestamp message [Po81c].

Various versions of NTP exist, beginning with the original RFC [Mi85], which included only data and packet formats, and the specification of the client/server engine. Version 1 [Mi88] added a self-organizing clock hierarchy and a logical clock algorithm, neither of which is integral to the protocol. Version 2 added authentication, control message capability, and asymmetric modes of operation, also supplemental to the protocol [Mi89b]. The latest NTP, Version 3 [Mi90b], includes an “overhauled” local clock algorithm, and new algorithms for peer offset combination; it also refines the definitions of delay, offset, and dispersion.

We use Version 3 NTP for our discussion and analysis. A summary of the version enhancements of NTP appears in Table 4.1. “Required” indicates whether the NTP specification denotes the component as required or optional, and “first version” indicates the version number in which the component first appeared.

NTP component	Required?	First version
packet exchange engine	y	0
logical adjustable clock	n	1
self organizing hierarchy	y	1
overcome unreliable	y	1
peer select / filter	n	1
control message	y	2
asymmetric modes	y	2
authentication	n	2

TABLE 4.1
NTP versions and components.

4.2. Casting NTP into Mirage

Viewing NTP using the Mirage model (‘casting’) involves describing the state space of NTP and interpreting messages of NTP in terms of the state space transformations of Mirage (Chapter 2). We define the state space of a protocol by the variables that characterize the protocol operation, rather than by those of the protocol specification. We differentiate the state that the protocol manipulates from the state required to manage that manipulation, in the sense of first-order state and second-order state. First order state characterizes the state space of the protocol, whereas second order state helps define varying characteristics of the transformation functions.

The Mirage model contains numerous references to time as absolute, so it may seem confusing to apply it to a protocol that manages clocks. Prior work in clock synchronization has called a clock “a function that maps real time to clock time” [Ma85], so the ‘time’ variable that NTP manages is just another linear state space. Clock protocols manage the agreement of a single variable (clock value), as it changes over time. Although time is used to manage the clock (e.g., in NTP using the polling interval), the effect of clock manipulations on the time variable of the protocol is ignored. NTP (and Mirage) assume that such variations are small in comparison to the intervals measured.

NTP endeavors to replicate a remote clock value that is assumed to be precise and accurate in its local space. The precision and accuracy of this clock are denoted by user classification, not within the protocol. The correspondence between the clock value and real time (true accuracy) is a semantic issue that is arguably not provable and that neither NTP nor Mirage proposes to address.

4.3. Resolution of domain differences

The Mirage model is designed for the domain where messages are delayed by a fixed, known amount, and the remote process exhibits variability in its computation. Conversely, NTP operates where messages are delayed by a variable amount (and potentially lost), and the remote process (clock) has very little intrinsic variability (error). We first describe how variability in message latency is equivalent to variability in computation at the remote node.

In NTP a round trip message exchange specifies a delay and offset; delay is the round trip message latency, and offset is the difference between the clock values. Consider the case where two nodes are absolutely correct, but the communication latencies vary. The true offset is zero, but the measured offset is not.

Let the unidirectional message delay be a probability density function (pdf) approximated by a Poisson distribution (Figure 4.5). The round trip delay is the sum of the unidirectional delays, and can be computed by the convolution of the unidirectional delay with itself (Figure 4.6). The offset is affected by the difference of the unidirectional delays, because outgoing delay causes the remote clock to be measured as late, whereas incoming (return) delay causes the remote clock to be measured as early. The difference in these delays is the convolution of the unidirectional delay with a reverse of itself ($f(-x)$),

as in Figure 4.7). Both these formulae assume that the unidirectional delays are identical pdfs.

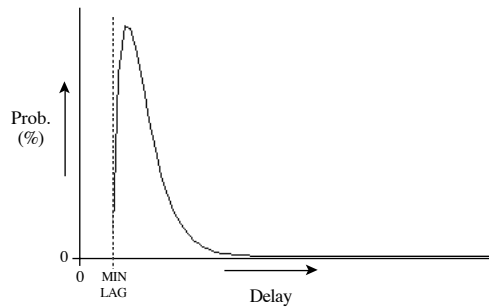


FIGURE 4.5
Unidirectional delay as a Poisson pdf¹

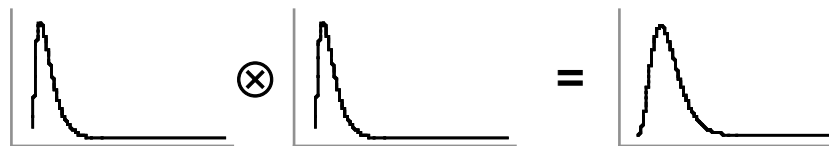


FIGURE 4.6
Bidirectional delay is the convolution of unidirectional delays

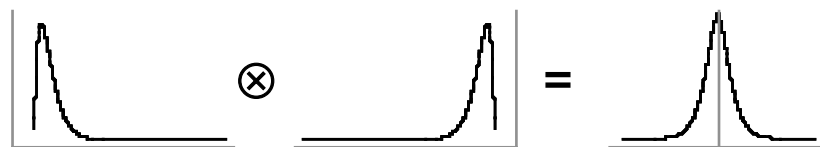


FIGURE 4.7
Measured offset is unidirectional delay convolved with its reverse

The resulting bidirectional delay has an Erlangian distribution (by definition³) (Figure 4.8). The measured offset exhibits the effects of the difference between incoming and outgoing delay, and that even perfect clocks are measured as statistically variable

¹probability density function.

²probability density function.

³An Erlangian distribution is defined as the convolution of a fixed number of identical Poisson distributions. In a queuing network, if each stage is Poisson in processing delay, then a path through the network exhibits Erlangian delay.

when communication latencies vary (Figure 4.9). The consequent offset pdf is symmetric about the zero offset, and appears Gaussian in form. An asymmetric pdf would result if the communication latency were asymmetric (forward vs. reverse paths).

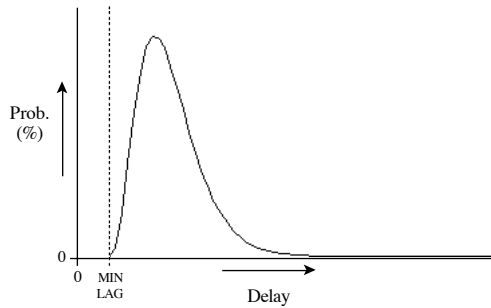


FIGURE 4.8
Bidirectional delay as Erlangian ($N=2$)

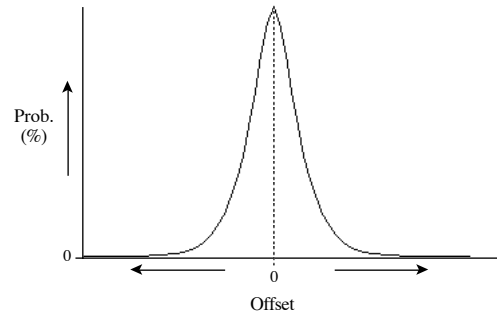


FIGURE 4.9
Measured offset as pseudo-Gaussian.

The bidirectional delay and measured offset of this system can be combined by an outer product, to show one possible structure of a delay/offset pair (Figure 4.10). The internal structure of such a graph cannot be determined from the delay and offset pdfs alone, because two one-dimensional pdfs underspecify the 2-dimensional delay/offset pdf. Even so, the outer product shows a structure similar to that of real NTP measurements, shown later. The result is that, even in the case of perfectly aligned clocks, a wide distribution of offset values occurs.

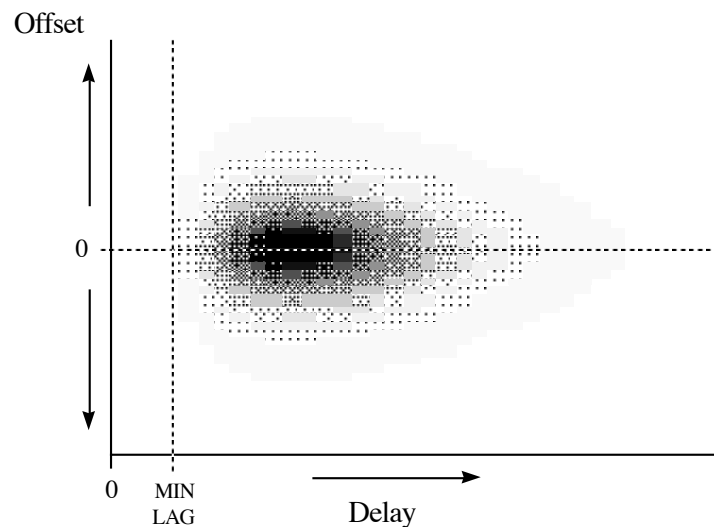


FIGURE 4.10
Probability vs. [delay, offset] pairs, density plot.

4.3.1. Analysis of delay and offset measurements

To verify these assumptions, we performed some measurements of NTP. We used a UNIX¹ shell script to exchange NTP packets (with timeout) between a local host (in Pennsylvania) with a remote node in California (a Stratum 2 NTP server). Each set of measurements reflects 2200 NTP requests (with a failure rate of approximately 1.5%, or 33 lost requests), at a rate of approximately 200 per hour, for a total of 11 hours, beginning at the time indicated. “Friday 8am” indicates a peak period (Friday 8am – 7pm EDT), and “Tuesday 8pm” indicates an off-peak period (Tuesday 8pm – 7am EDT).

The offset curves are Gaussian-like (Figures 4.11, 4.12), in which the peak measurements (Figure 4.12) have *slightly* higher variance than off peak (9.3% higher variance for peak).

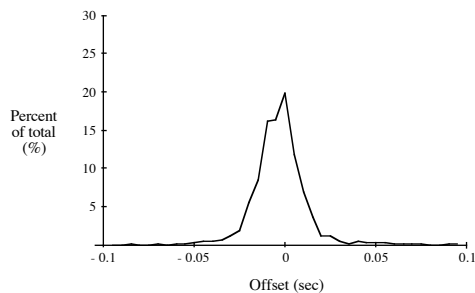


FIGURE 4.11
Tuesday offset values (off-peak)

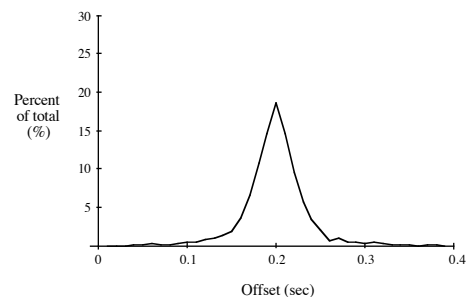


FIGURE 4.12
Friday offset values (peak)

The delay measurements are also as expected, approximating Erlangian distributions (Figures 4.13, 4.14). The off-peak measurement exhibits a smaller minimum delay (Figure 4.13). Both experiments show an unexpected quantization, especially because it appears in the delay measurements but not in the offset measurements, and both quantities are derived from the same timestamp sets (from NTP message headers).

The quantization may be the result of using a fixed set of IP paths, except that the quanta are nearly exact units of 0.01 seconds (10 ms). We assume that this quantization was due to a fixed service interval in the local NTP server response loop. This conclusion is further complicated by the fact that some replies have delay quantizations that are slightly smaller than the 0.01 second quanta. Similar quantizations are manifested in ensemble averages, so we conclude that the code of our local NTP implementation is the

¹UNIX is a registered trademark of Unix Systems Laboratories.

most likely suspect. Further analysis of NTP may prove this conclusion, but is beyond the scope of this dissertation.

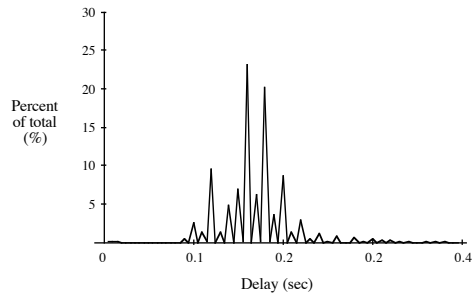


FIGURE 4.13
Tuesday delay values (off-peak)

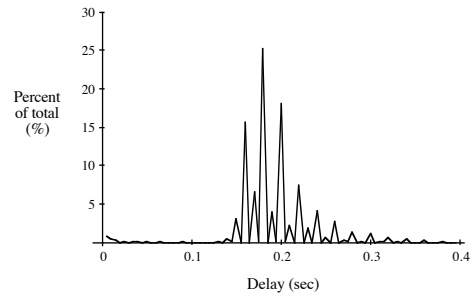


FIGURE 4.14
Friday delay values (peak)

Regardless of the quantization effects, the curves are bounded by Erlangian-shaped envelopes. Again the peak values have a slightly higher variance (15.7% larger variance for peak), although this is because the peak delay pdf has a longer ‘tail’ of values of higher latency. The peak pdf is has a larger minimum delay, and delay values are grouped more tightly about the mode than in the off-peak. The higher latency is the result of higher network loads during peak times. The more tightly grouped latency values are most likely the result of side effects of network load on routing information accuracy.

The NTP implementation we tested is supported by the IP datagram transport protocol. IP routing information is maintained as a side effect of IP datagram transmission [Co91a]. Higher network loads result in more accurate routing information, which in turn results in more precise (i.e., repeatable) latency. Lower network load can permit routing table inaccuracies to persist longer, so that datagrams experience larger variability in route paths, in turn increasing delay variability.

These measurements indicate that: (1) there is a minimum delay that is not affected by the latency variability, so there exist tighter bounds on the measured offset, and (2) forward and reverse path delays are not equal. The distributions of delay/offset pairs for both peak and off-peak data sets are shown in Figures 4.15, 4.16, respectively. The previous plots of offset and delay are the (respectively) horizontal and vertical projections of these plots (Figures 4.13, 4.14, 4.11, 4.12). For comparison, we also present a 3-dimensional plot of the off-peak values (Figure 4.17).

The round trip latency can be partitioned into two components – a fixed minimum, and a variable additional delay, where each component is non-negative (by definition).

The corresponding fixed minimum components of the two alternate bounds are plotted as vertical lines (gray and dashed).

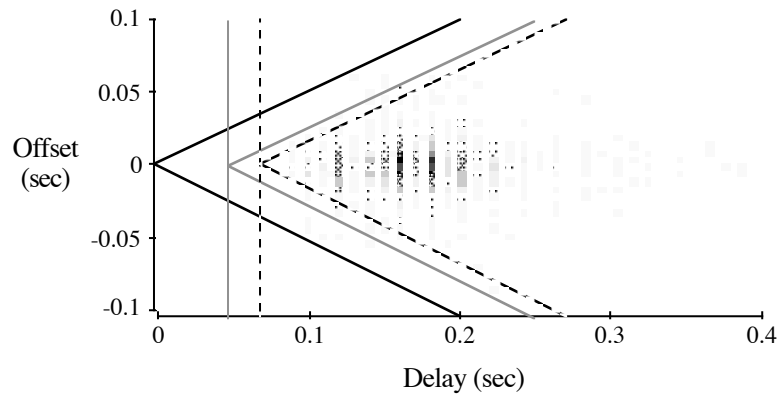


FIGURE 4.15

Delay v.s offset, time-repetition density (off-peak)

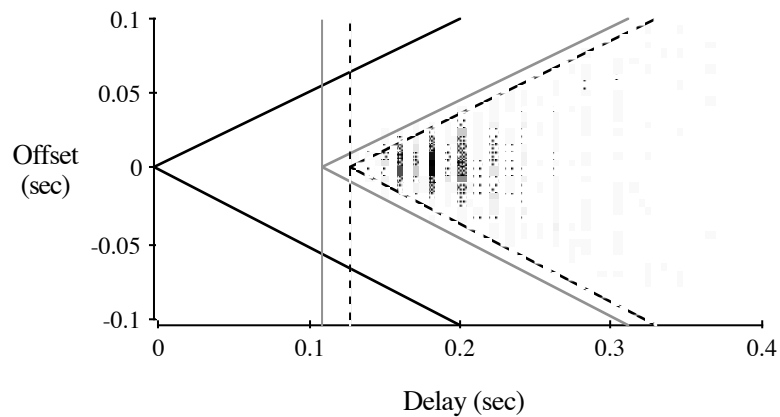


FIGURE 4.16

Delay vs. offset, time-repetition density (peak)

In these plots, the NTP bounds (Equation 4.8) are shown as solid diagonals. These bounds appear too loose, and are based on the assumption that one-way latency is bounded by the total round trip latency only. Shifting these bounds to the right yields more appropriate boundaries to the displayed data; two such alternate bounds are shown (loose in gray diagonal, and more restricted in dashed diagonal lines). These alternate boundaries suggest reinterpretation of the bound on the difference between outgoing and incoming latency, and their effects on the measured offset values.

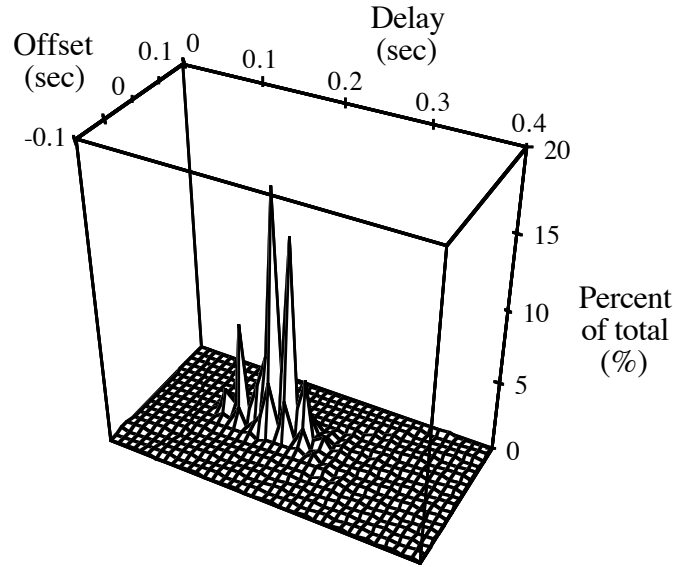


FIGURE 4.17

Delay vs. offset vs. probability, time-series (off-peak)

Off-peak measurements have larger minimum delay components, but there is less variability in our visual estimation of the minimum delay, compared to peak measurements. Again, this can be explained as an effect of the routing table maintenance, where light network load results in low delay, but in high variability in delay, because route information is updated less often. High load causes high delay, with low variability because heavy traffic helps maintain accurate route information.

The partitioning of round trip latency into non-negative fixed and variable components (Equation 4.9) can be incorporated into the previous constraint equations (Equations 4.7, 4.8), resulting in Equations 4.10, 4.11.

Equation 4.9: $\delta_{total} = \delta_{fixed} + \delta_{var}$

where $\delta_{fixed} \geq 0$ and $\delta_{var} \geq 0$

Equation 4.10: $0 \geq \delta_{uni-var} \geq \delta_{var}$

Equation 4.11: $\theta + \frac{\delta_{var}}{2} \geq \hat{\theta} \geq \theta - \frac{\delta_{var}}{2}$

Visually, the effects of this partitioning of the delay can be shown by modifying the NTP message exchange diagram (Figure 4.2) to Figure 4.18. Black portions indicate fixed components of the delay, and gray indicates variable components; the remote interval is bounded between the fixed components of latency. The bounds on the offset are more tightly restricted as a result (Figures 4.19, 4.20), compared to the earlier diagrams (Figures 4.3, 4.4).

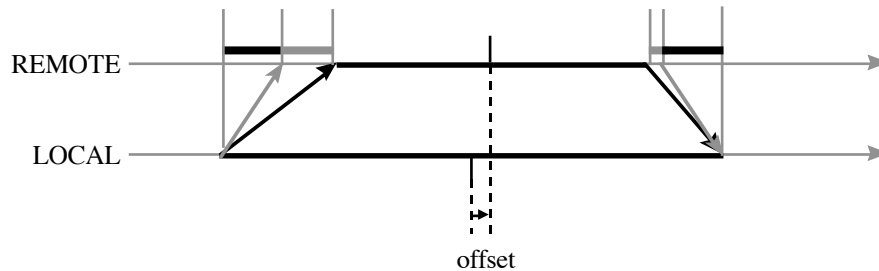


FIGURE 4.18

Fixed (black) and variable (gray) delay in message exchange

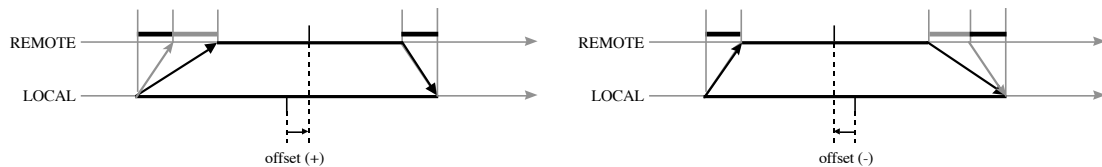


FIGURE 4.19

Exchange maximum offset, variable delay

FIGURE 4.20

Exchange minimum offset, variable delay

Thus far the delays have been assumed identical on the forward and reverse path of a message. If this were the case, all measurements would exhibit a symmetric offset distribution. Temporal data sets do exhibit this behavior, but ensemble data sets are heavily skewed toward positive offsets (Figure 4.21). This skew is the result of a larger average forward latency, which is the result of routing effects in the underlying IP transport mechanism.

An ensemble data set consists of a set of unique message destinations. Each message is routed to the destination and back. The reverse path of a message occurs by an updated and presumably shorter path than the forward message, because the forward message updates routing information as it is delivered. Forward messages are routed by comparatively stale information, exhibiting longer routes as a result.

Lower strata (e.g., Stratum 1) servers exhibit large offset skews, which disappear as the stratum increases (Figures 4.22, 4.23, 4.24, 4.25)¹. Offset skews should tighten as strata decrease due to higher precision in local clocks, whereas empirically, offsets tighten and become more symmetric as strata increase. Both the tightening and more symmetric nature of higher strata can be attributed to the locality (and thus known routes) of lower strata servers, whereas higher strata servers were typically more distant; this is a side effect of the NTP server hierarchy and its partitioning of siblings according to their parent and topological access criteria. Empirically, lower strata servers were more distant than higher strata servers.

Figure 4.21 also includes NTP bounds (black diagonal lines), violations of the NTP bounds (circles), and a possibly more constrictive bound (gray diagonal lines) resulting from an assumed latency minimum (gray vertical line).

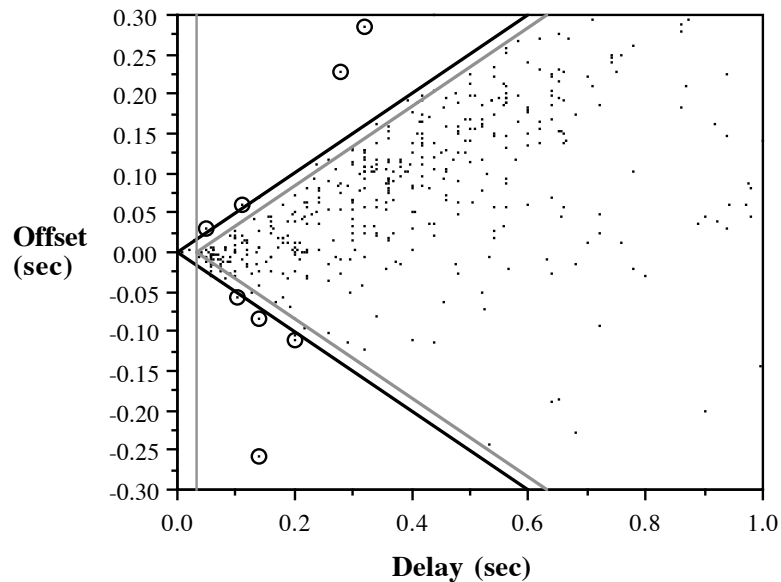


FIGURE 4.21

Delay vs. offset, entire ensemble (all strata)

¹Stratum 0 server replies are not shown; only 5 were accessible, and '0' indicates an unknown stratum.

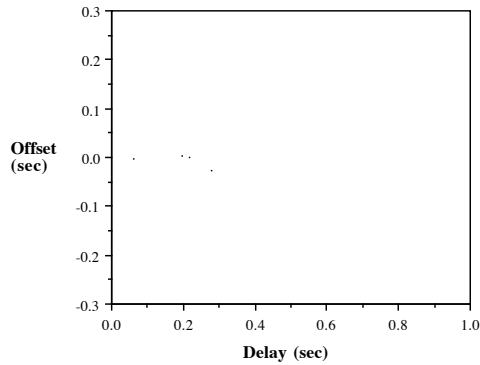


FIGURE 4.22
Stratum 1 ensemble

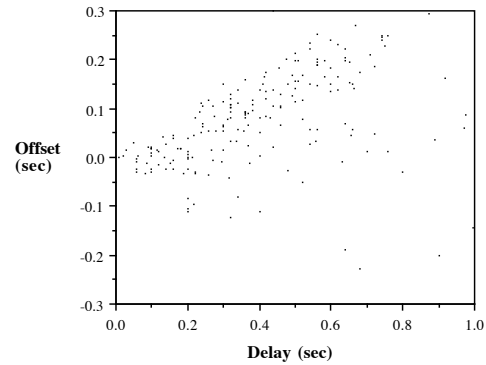


FIGURE 4.23
Stratum 2 ensemble

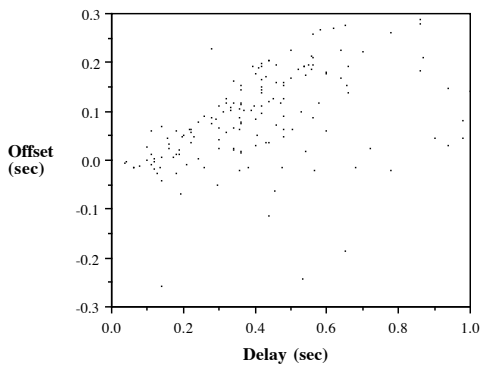


FIGURE 4.24
Stratum 3 ensemble

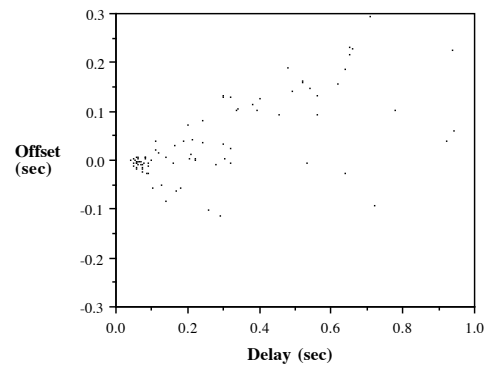


FIGURE 4.25
Stratum 4 ensemble

4.4. Description of NTP in Mirage

The following is a description of the variables of NTP, (some individual, some groups), and their description as Mirage indicates (Table 4.2). *Local state* denotes the protocol state space as Mirage considers it, and *remote perception* is the value obtained by communication with a remote node, i.e., part of the node's total view. *Computation function parameters* denote those components of NTP that are part of the Mirage time-

transformation function. *Protocol management* denotes components that affect the boundaries of the computation function, and that govern the send actions. These last management operations are part of communicability, and the optimization thereof.

NTP 'state' ¹	NTP description	Mirage description
(time)	current clock value	local state
(frequency scale)	clock frequency adj.	comp. func. params
drift	frequency variability	comp. func. params
wander	drift variability	comp. func. params
offset	remote clock shift	remote perception
delay	packet RTT	protocol mgt.
dispersion	variance in packet data	protocol mgt.
elapsed time	time since last request	protocol mgt.
poll interval	time between requests	protocol mgt.

TABLE 4.2

Mirage interpretation of the state variables of NTP

4.4.1. State space

The local state of NTP consists of the state of the local clock, i.e., the current time. The local perception of a remote clock is the offset, i.e., the difference between the remote clock and the local clock. The remainder of the state variables of the NTP specification refer to protocol management or expression of computability (to describe the time transformation). The only variable that is managed by NTP is time; all others serve to facilitate this management.

¹Parentheses indicate variables which are not listed as explicit in the NTP protocol description, but are implicitly required.

4.4.2. Transformations

Transformations of the state space of NTP, as described by Mirage, map sets of time values onto other sets of time values. These sets affect the perceptions of remote clocks. In NTP, sets of time values are denoted by continuous intervals, so transformations map time intervals to other time intervals.

Some components of NTP resemble those required in Mirage to manage the state space and maintain stability. *Elapsed time* indicates the time since the last clock update, and indicates when stability will fail. This, together with the assumption of a fixed skew (0.01 sec/day), indicates the current imprecision of the clock (*dispersion*), and is used to determine the expansion of the time transformation. The *poll interval* indicates the time between messages, and is a precomputed expected time when the state becomes imprecise enough to require a message initiation, i.e., the maximum interval over which stability can be maintained.

4.4.2.1. Time

The local clock of a node changes as a function of time, both by tracing a path of sequential clock values (ticking), and by becoming less precise as time progresses (in the absence of received messages). The ticking of the clock is modeled by Equation 4.12 (depicted in Figure 4.26), from the definition of frequency as the first derivative of the clock transition function, drift as the second derivative, wander as the third, etc., and includes noise.

The imprecision of the clock is modeled by Equation 4.13. The imprecision describes the expansion of the state point (clock time) to a state interval (time +/- imprecision), where the distribution within this interval is assumed to be uniform (Equation 4.15, Figure 4.27).

Equation 4.12:
$$T(\Delta t) = \text{noise}(\Delta t) + \text{epoch} + F(\Delta t) + \frac{1}{2}D(\Delta t^2) + \frac{1}{6}W(\Delta t^3) + \dots$$

Equation 4.13:
$$\text{imprecision}(t) = \{\text{bit_precision}\} + \text{max_skew_rate} * t$$

Equation 4.14:
$$T(t) = \text{noise}(t) + T(t_0) + R(t_0)[\Delta t] + \frac{1}{2}D(t_0)[\Delta t]^2$$

Equation 4.15: $T_interval(t) = [T(t) - imprecision(t), T(t) + imprecision(t)]$

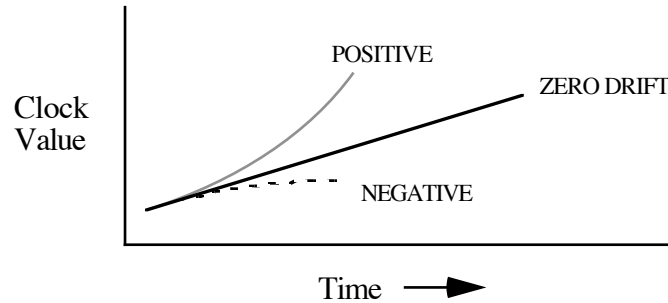


FIGURE 4.26

Clock value is a function of time

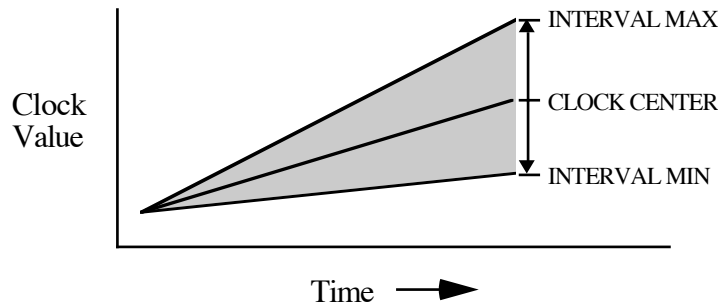


FIGURE 4.27

Clock interval as fixed expansion centered on time function

In NTP, wander and higher order variability effects are not considered. The combination of the equations for time transformation and imprecision growth result in the same formula as in NTP (Section F.3 of [Mi90b]), which describes the operation of the logical software clock (Equation 4.14). The imprecision of the clock is a fixed, linear function of elapsed time since the last clock adjustment, and is not determined by clock precision, strata, or previous adjustments.

The software clock model of NTP is called supplemental and external to the protocol in its specification. We disagree, because the clock equations are required to specify the time transformation of the Mirage model of NTP.

NTP makes the same assumptions as Mirage, that although ‘time’ can vary as the result of the protocol, variability in the progression in time is small enough that it does not affect the model substantially, i.e., it is not contradictory to model displayed time against a virtual absolute timescale (even if the latter is unknown).

NTP further describes the noise function as a pdf, in which a single initial clock value spreads into a zero-centered bell-shaped curve (Gaussian) of the pdf. This curve occurs as the vertical cross section of the time transformation graph (Figure 4.27), so that the figure has a bell-shaped surface in the third dimension (not shown).

There are also several constraints on the clock function. Clocks never run backwards, so the drift parameters can never combine to cause reversal of the clock. The clock distribution is similarly limited, so that the slope of the interval minimum line can never be negative.

4.4.2.2. Send

The effects of a sent message are not considered in NTP. The protocol is intended to create a hierarchical organization of clients and servers, where clients are served by their parents in the hierarchy. As a result, clocks are modeled unidirectionally, i.e., the child models the parent only. The parent does not model the child.

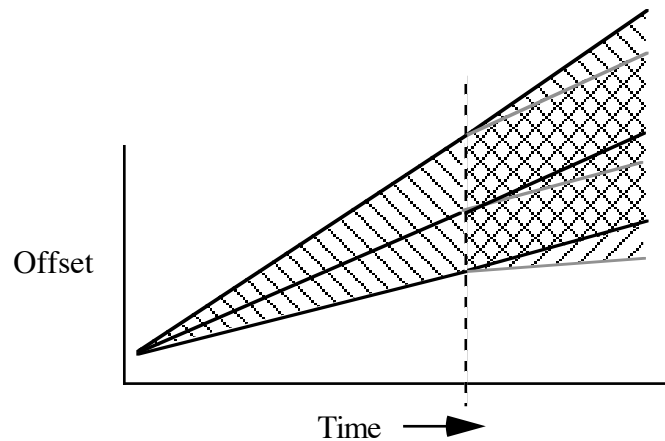


FIGURE 4.28

Transformation of a perception due to a sent NTP message

We can extrapolate our investigation to the case where the parent would model the child. Emitting a time message would cause an adjustment in the center-line of the expanding state (Figure 4.28). The resulting state space consists of the union of the

affected and unaffected expanding state spaces, indicating an increase in imprecision caused by the adjustment. The parent node doesn't know whether the sent time is accommodated into the child node, so it adjusts its perception based on the new state emanating from any existing possible clock. The result is an adjustment of the clock frequency only.

4.4.2.3. Receive

Received messages alter the local state in NTP, as proscribed by Mirage. A received message causes a node's perception of the remote state to be updated, which collapses the interval to a point, from which subsequent expansion will resume (Figure 4.29). Other information is also extracted from this collapse.

If the received time is in the upper region of the interval, then the local model of the remote clock is slow (the expanding triangle is angled too far downward), and needs to be compensated by shifting the triangle up. The shift in the triangle is reflected in an update of the frequency parameter of the computation function.

From this analysis, we conclude that the components of the computation function, i.e., frequency, drift, etc., are not constants, as initially perceived, but variables in the model which need to be incorporated. There is an interaction in these equations that indicates that the variables are not an orthogonal basis of the system space. The result is a revised set of time transforms (Equations 4.16, 4.17, 4.18), plus a set of receive transforms (Equations 4.19, 4.20, 4.21).

The receive transforms use the delay and offset information derived from the received message, and assimilate it into the local time view as specified by recombination algorithms in the protocol. NTP does not manage multiple perceptions within a single node; an ensemble data set is combined using the peer selection algorithm. Similarly, a temporal data set is collapsed to a single value by the data filter algorithm. This is required to reduce the computation overhead in resolving a set of clock values to an indicated correction, and to reduce the storage required over time to maintain data of previous state.

Note that both the peer selection and data filter algorithms are integral to the Mirage model of NTP, because they specify the receive transformation operations, whereas the NTP specifications refer to the equations as optional "suggested implementations".

The slope of the expansions is indicated by the error function (Equation 4.22). NTP uses fixed values for the skew, currently set to a constant of 0.01 second/day. In Mirage,

this corresponds to the computation function, and need not be linear in the elapsed time since last update. In NTP, time updates never alter the value of the expansion, only the center-line of it.

$$\text{Equation 4.16: } T(\Delta t) = \text{noise}(\Delta t) + \text{epoch} + F(\Delta t) + \frac{1}{2}D(\Delta t^2) + \frac{1}{6}W(\Delta t^3) + \dots$$

$$\text{Equation 4.17: } F(\Delta t) = F(t_0) + D(\Delta t) + \frac{1}{2}W(\Delta t^2) + \dots$$

$$\text{Equation 4.18: } D(\Delta t) = D(t_0) + W(\Delta t) + \dots$$

$$\text{Equation 4.19: } T(\Delta t) = \{\text{weighted avg. of clocks}\}$$

$$\text{Equation 4.20: } F(\Delta t) = \{\text{exponential avg. of clock differences}\}$$

$$\text{Equation 4.21: } D(\Delta t) = \{\text{current freq.} - \text{avg. freq.}\}$$

$$\text{Equation 4.22: } \text{error}(\Delta t) = \{\text{bit_precision}\} + \text{max_skew_rate}(T_{in} - T_{orig})$$

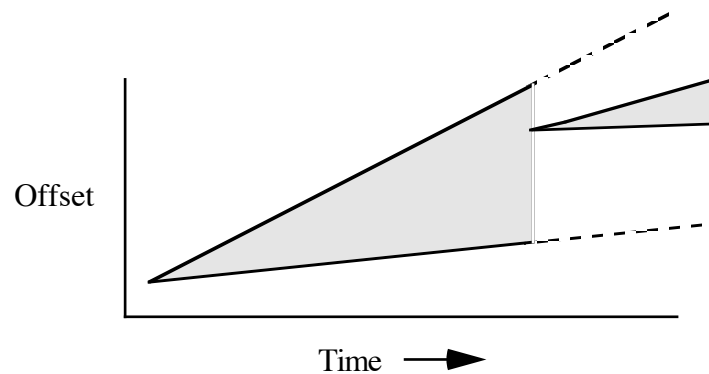
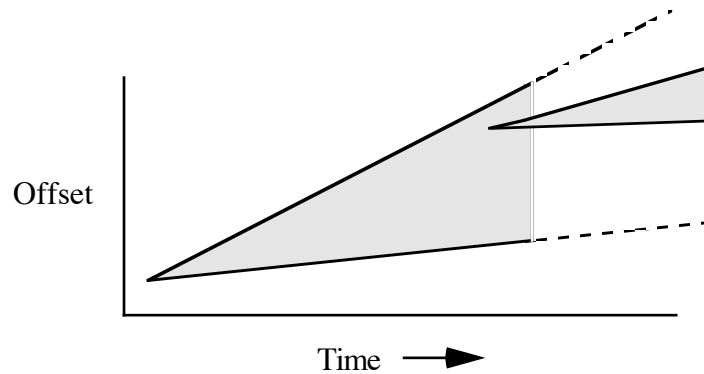


FIGURE 4.29

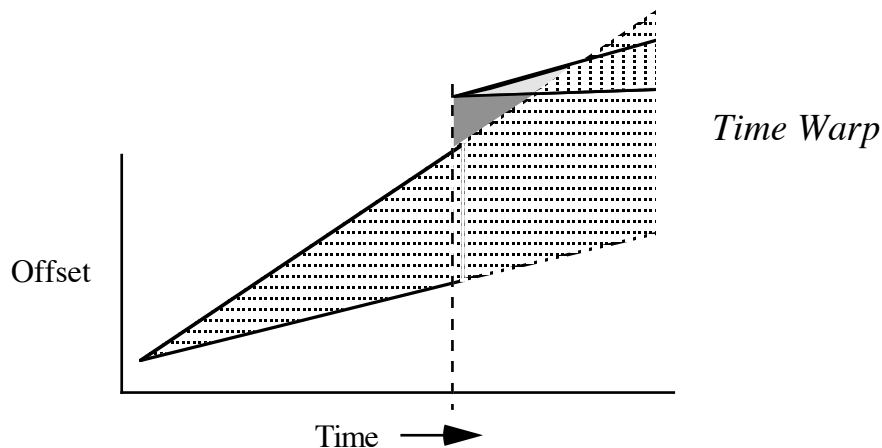
Transformation of a perception due to a received NTP message

As noted before in the Mirage model description (Chapter 2), the delay in a message's transit since its origination must be measured, and the current state must be back-calculated from it (Figure 4.30).

**FIGURE 4.30**

Receive transformation, accommodating transit time effects

As also noted in the Mirage model, there are restrictions on the collapse of the state space, as indicated by the received message. If the message indicates that the new state space value is *not* contained in the current model of the remote node, then an inconsistency exists. NTP calls this type of inconsistent received message a *Time Warp*, and it correlates to Mirage receive constraint criterion violation (Figure 4.31). Specifically the violation occurs in NTP because the received offset cannot be incorporated within a continuous valid interval, whereas in Mirage the violation occurs because the collapsed state is not a member of the set of current possible states.

**FIGURE 4.31**

Representation of a Time Warp in the state space timeline

4.4.3. Partitioning of the state space

In NTP, the state space is modeled as a continuous interval of possible clock values. In Mirage, the bit size of the message guard indicates the extent of the partitioning of this interval (here we assume an equipartitioned interval).

NTP does not send guards on the messages; they are inferred from the relative position of the received clock adjustment to the computed valid interval. If the received message is in the upper portion of the interval, the current clock frequency is low (i.e., the clock is slow), and it needs to be speeded up.

If the interval is not partitioned, received clock adjustments result in new clock values, but frequencies, drifts, etc., are not compensated (Figure 4.32, 0 bits). If the interval is partitioned into two (i.e., as if a one bit ‘virtual’ guard is inferred), then the frequency is adjusted faster or slower, but only by a fixed amount in either case (Figure 4.32, 1 bit). If the interval is partitioned into progressively smaller components, more accurate adjustments of the frequency can be made (Figure 4.32, 0-4 bits shown). NTP never alters the drift value.

As a result, NTP uses ‘virtual’ guard information to adjust the clock by a value indicated by a partition inversely proportional to the number of unique frequency adjustment parameters. Guards are inferred from clock timestamp information in the NTP messages, which are much larger than the required guard size.

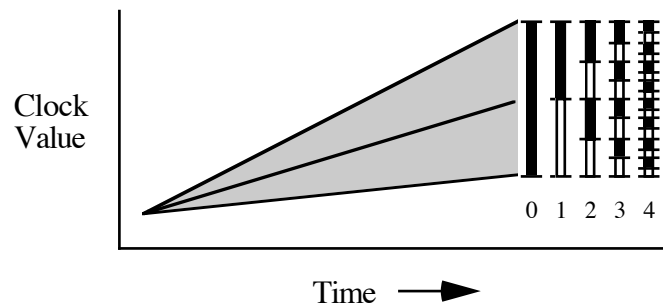


FIGURE 4.32

Partitioning of the state space results in variably ‘tight’ state collapse

Another interpretation of the partitioning is the way in which larger messages specify more precise clock update values. If no bits are sent, the clock is not updated. If one bit is sent, then the clock is in one of two indicated subintervals of the currently valid

interval. The more bits that are sent, the more precisely the receiver can update the interval (again, Figure 4.32).

In either case, the guards are virtually specified by the sent clock value. The larger the guard, the more finely partitioned the state space (interval) becomes, and the more precisely the space can be maintained. The problem is that larger guards also increase time between sent messages, because larger guards require longer messages. Consider the case where messages specify clock values to within 10 ms, and take 1 minute to send, vs. messages that specify the clock value to within 1ms, and take 1 hour to send. If the clock skew rate (computation expansion function) is small (less than 1ms/hour), then the long, (infrequent) precise messages are more effective in synchronizing the clock than short, (frequent) imprecise messages. When the clock skew rate is large (more than 60 ms/hour), the short, (frequent) imprecise messages can maintain the clock to within 60ms/hour, whereas the long messages also let the clock skew out of adjustment by a larger amount, even though it may be more tightly synchronized for a short time immediately after being adjusted.

4.4.4. NTP degenerates to a clock pulse

NTP clock value messages degenerate to a clock pulse (hardware clock signal) under certain conditions. If the latency variability is zero, the clock is completely described by the temporal transition function. If drift and other higher order components are zero, and if the frequency component of the clock is zero, the clock is described by a perfect line. If frequency error is not zero, there is an expansion of the interval with time. The sender limits the extent of this error by the frequency with which adjustments are sent. The remote clock is partitioned into two components (time received, time not received), so reception of any unique signal suffices to manage the clock. This is a regular clock pulse, because the frequency error in the receiver is the size of the true frequency (because the transition function assumes a static clock value (i.e., zero frequency) with an error that increases with time (Figure 4.33). The period of sent messages determines the extent to which the sender's clock is accurately modeled, and the centerline of the sent messages expresses the clock frequency.

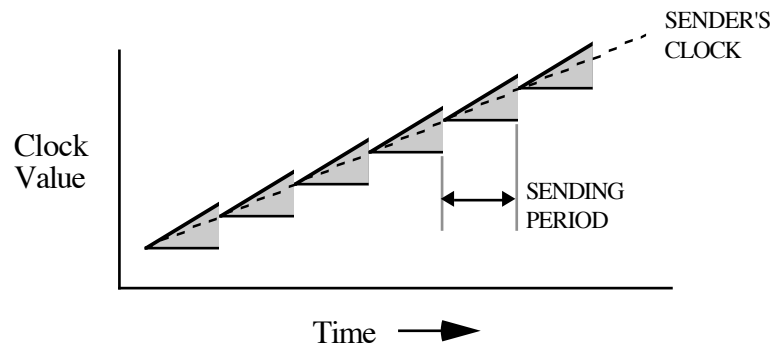


FIGURE 4.33
Regular sender anticipation.

4.4.5. Constraints

Many of the constraints in the Mirage model overspecify components of the NTP protocol. For example, the message size constraints of Mirage indicate that smaller messages could be used in the NTP header, where in the latter a timestamp indicates a 200 picosecond interval within a 136 year span. The overspecification of the timestamps is used by NTP for other consistency checks, e.g., to ensure that incorrect timestamps do not ‘wrap around’ onto valid stamp values. Consistency checks include measuring positive intervals for client/server responses, verifying reasonable delays, and possible formal authentication of the packet using an auxiliary mechanism.

4.5. Observations

The main observation from the application of the Mirage model to NTP is that NTP makes little use of Mirage components. NTP has a very simple state (time, frequency, drift) that is governed by simple (quadratic) equations, and has a fixed linear computation function. Receive transforms are elaborate, but only insofar as they accommodate multiple Byzantine sources in a way that attempts to discern true time from a perception (through peer selection and filter algorithms); Mirage is intended to maintain the perception only. NTP uses intricate weighting functions to collapse the state space which Mirage preserves (inefficiently, in comparison to NTP).

We have observed an isomorphism between variability in latency, which NTP accommodates, and variability in the computation expansion, which Mirage accommodates. Certain characteristics of the delay and offset measurements in NTP can be predicted from simple models of the network characteristics (Poisson unidirectional latency), with the “Mirage function” embodied as the convolution of the component pdfs.

We were previously unaware of the extent to which Mirage relies on accurate time measurements (relative time), especially in the computation functions. This was however no more so than other protocols similarly rely, even clock synchronization protocols.

Some constraints of Mirage are unnecessary in the NTP protocol, notably those indicating the bit size of the sent and received messages, because NTP uses fixed headers with oversized messages intended to provide informal authentication and error resilience. Other constraints specified in Mirage also appear in NTP, e.g., an NTP *Time Warp* is a violation of the state space collapse being a subset of the existing state space model.

NTP circumnavigates issues of modeling sent messages, as well as sender anticipation, by organizing the time server network into a strict hierarchy. We can add sender anticipation to NTP, which results in a server-initiated periodic ‘chime’, similar to NTP’s *broadcast* mode of operation.

4.5.1. Gains

NTP uses a fixed header description, with fixed message sizes. The message sizes can be reduced to minimal values as specified in Chapter 2. The sizes of the messages are a function of the variability in the remote state, which in NTP is isomorphic to variability in the round trip time, as described earlier. As the variability in round trip time decreases, the need for extended precision clock stamps in the header is reduced; a zero round trip variability indicates that the clock discrepancies are a function only of local drift and error. As local drift and error approach zero, there is less need for clock value correspondence. Zero round trip variability and zero drift and error would abolish the need for communication altogether.

Note that these conclusions require that latency variability and drift disappear; actual round trip latency does not matter, so long as it can be predicted with a probability of 1.

4.5.2. Prior work

Relevant prior work on the analysis of time protocols includes probabilistic clock synchronization [Cr89], optimal clock synchronization [To87], and distributed clock mechanisms [Ma85].

All of these analyses rely on bounds on the maximum round trip latency to determine achievable precision and accuracy, according to the same bounding formula as in NTP (Equation 4.8). A smaller round trip latency results in less error in the computed offset. We have shown that offset error is linearly proportional to the variability in round trip latency, and more precisely to smaller difference between forward and reverse path latency (Equation 4.11).

We do not consider Byzantine failures, or the explicit failure of any components of the protocol in the Mirage analysis. Mirage shows the errors in the perception of remote clocks, even where the remote clock is perfect. Byzantine and fail-stop failures would increase the error in the local perception of the remote clock, although determination of whether stability could be achieved is beyond the scope of this analysis. We limited this analysis to the characteristics of NTP.

The methods used here are similar to those used in statistical timekeeping analysis [Cr89], especially those relating the pdf of latency values to a time protocol. Here we additionally show the way in which a Poisson unidirectional latency variability can cause Erlangian round trip variability, and Gaussian-like measured offset variability, even in the presence of a perfect remote clock (i.e., one with no temporal expansion in state).

The statistical method suggests increasing the number of message attempts to increase the probability of receiving a message with a minimal latency, potentially increasing the link latency due to load increases [Cr89]. Other methods use sets of messages to overcome node failure [To87] or ensure accuracy [Ma85], but indicate nothing of the tradeoff between message size and frequency. These are self-defeating mechanisms, which Mirage counteracts by indicating that more frequent communication allows use of smaller guards and smaller messages.

Mirage does not claim optimality in the number of messages exchanged alone; rather, it optimizes the total communication in message number and length (integrating length over the message set), and considers optimality only insofar as the desired stability can be achieved and maintained. Furthermore, [Cr89] permits failure of the protocol to achieve the desired precision, thus permitting instability, which Mirage prohibits.

4.5.3. Conclusions

The use of Mirage to analyze NTP has highlighted aspects of NTP not required in the specification, and helped refine our understanding of Mirage. NTP specification considers the logical clock, peer selection, and filter algorithms optional to the protocol specification. Mirage cannot model NTP without the use of constraints from these algorithms, so we consider them integral components of the NTP protocol, and would suggest that they be required in the specification.

We showed the equivalence between latency variability and imprecision in the local perception of a remote state. We concluded that the NTP protocol boundaries on measured offset error can be corrected, to denote explicitly that measured error is the result of latency variability alone. This results in more accurate constraints on the measured offset error.

Using message length and frequency tradeoffs in the Mirage communicability formula (Chapter 2), we show the degeneration of NTP to a hardware clock signal, as latency variability approaches zero. The Mirage was used to derive exact effects of latency variability on measured offsets, by applying the discrete Mirage formula to continuous latency distributions using pdf convolutions.

Before this investigation, we were not aware the extent to which the Mirage model relies on time measurements for its analysis. This use is not precluded by the application of Mirage to even clock synchronization protocols, because the assumptions of measurement required within Mirage are similar to the requirements in other clock synchronization model analyses.

NTP was useful as a sample protocol for Mirage analysis, but many of the more interesting components of Mirage were not applicable to NTP. Messages in NTP are static and overspecified as required by Mirage's message constraints. NTP models time as continuous intervals, whereas Mirage permits arbitrary sets in its model. Sender anticipation and guarded messaging can be inferred from some aspects of message processing in NTP, but are not explicitly applicable to this protocol. Finally, some of the most interesting algorithms in NTP, those of peer selection and data filtering, are mechanisms to collapse and project multiple dimensions of the state space that Mirage is intended to preserve.