Distributed computing
“Time synchronization”

Mathieu Delalandre
University of Tours, Tours city, France
mathieu.delalandre@univ-tours.fr

Lecture available at http://mathieu.delalandre.free.fr/teachings/dcomputing.html

Time synchronization

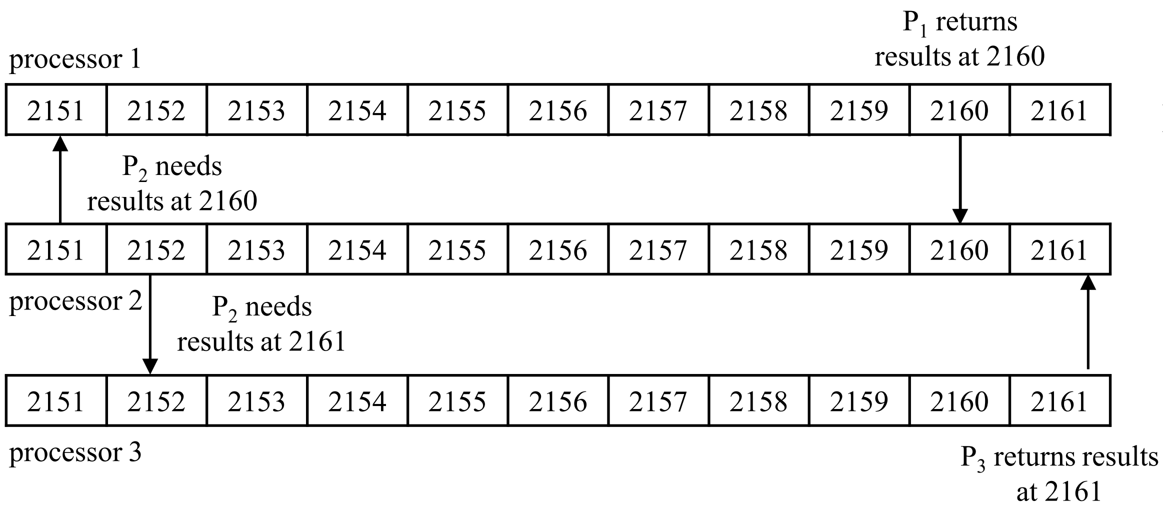
1. Introduction
2. Clock synchronization
    - 2.1. Coordinated Universal Time (UTC)
    - 2.2. Network Time Protocol (NTP)
    - 2.3. Global Positioning System (GPS)
    - 2.4. Berkeley Algorithm
    - 2.5. Reference Broadcast Synchronization (RBS)

Introduction (1)

Causality: two events are causally related if the nature of behavior of the second one might have been influenced in any way by the first one. Causality among events is a key concept in programming for a variety of problems: compilation, database systems, web browsers and text editors, secure systems, fault diagnosis and recovery, etc.

Within a centralized multi-processor system, the clock is common to all the processes or synchronized, IPC is immediate due to synchronization mechanisms. As a consequence, execution is **synchronous** and causality can be easily implemented.

e.g. 3 processes P1, P2, P3 on 3 processors on a same computer considering that P1 must return results to P2 before P3 returns results to P2.



A synchronous execution is an execution in which:

(i) processors are synchronized and the clock drift rate between any two processors is null or bounded,

(ii) message delivery times are such that they occur in one logical step or round,

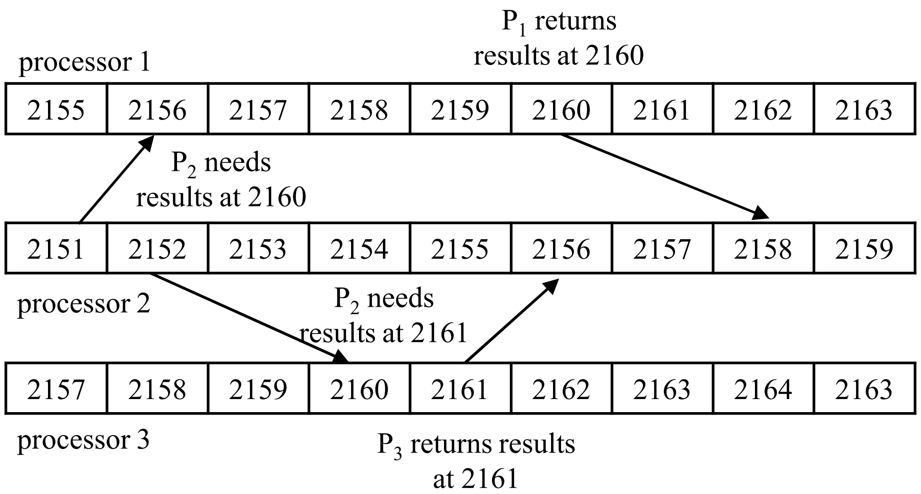
(iii) we can have a known upper bound on the time taken by a process to execute a step.

Introduction (2)

Causality: two events are causally related if the nature of behavior of the second one might have been influenced in any way by the first one. Causality among events is a key concept in programming for a variety of problems: compilation, database systems, web browsers and text editors, secure systems, fault diagnosis and recovery, etc.

Within a distributed system if the physical clocks are not precisely synchronized, the causality relation between events may not be accurately captured. As a consequence, execution is mainly **asynchronous**.

e.g. 3 processes P1, P2, P3 on 3 processors on different computers considering that
P1 must return results to P2 before P3 returns results to P2.



An asynchronous execution is an execution in which:

(i) there is no processor synchrony and there is no bound on the drift rate of processor clocks,

(ii) message delays are finite but unbounded,

(iii) there is no upper bound on the time taken by a process to execute a step.

Introduction (3)

Clock synchronization is the process of ensuring that physically distributed processors have a common notion of time. When clock are synchronized, it supports distributed synchronous execution.

There are four main issues:

1. how to fix the reference time t?
   e.g. UTC, mean time, etc.
2. is the synchronization based on a active or passive server?
3. the considered network for the synchronization mechanism.
4. how to deal with the precision in synchronization
   e.g. message delay estimation, clock skew, etc.

Methods	**Reference Time**	**Server**	**Network**	**Precision**
Network Time Protocol	UTC	passive	Internet	estimation message delay
Global Positioning System (GPS)			UHF	clock offset estimation
Berkeley Algorithm	mean time	active	Ethernet	none
Reference Broadcast Synchronization (RBS)	clock offsets		Wireless	clock skew

Time synchronization

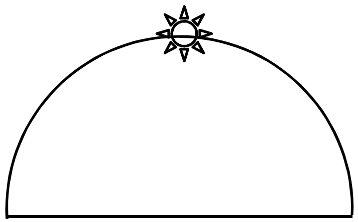
1. Introduction
2. Clock synchronization
    2.1. Coordinated Universal Time (UTC)
    2.2. Network Time Protocol (NTP)
    2.3. Global Positioning System (GPS)
    2.4. Berkeley Algorithm
    2.5. Reference Broadcast Synchronization (RBS)

Clock synchronization

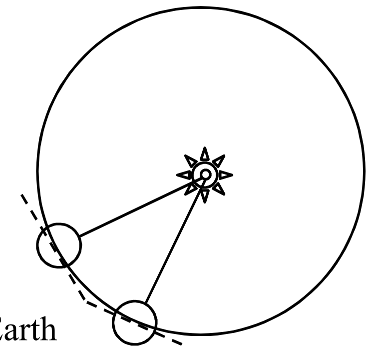
<table>
<tr><th>Methods</th><th>Reference Time</th><th>Server</th><th>Network</th><th>Precision</th></tr>
<tr><td>Network Time Protocol</td><td rowspan="2">UTC</td><td rowspan="2">passive</td><td>Internet</td><td>estimation message delay</td></tr>
<tr><td>Global Positioning System (GPS)</td><td>UHF</td><td>clock offset estimation</td></tr>
<tr><td>Berkeley Algorithm</td><td>mean time</td><td rowspan="2">active</td><td>Ethernet</td><td>none</td></tr>
<tr><td>Reference Broadcast Synchronization (RBS)</td><td>clock offsets</td><td>Wireless</td><td>clock skew</td></tr>
</table>

Clock synchronization
"Coordinated Universal Time (UTC)" (1)

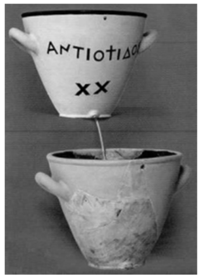
Transit of the sun occurs when the sun climbs to a maximum height in the sky.

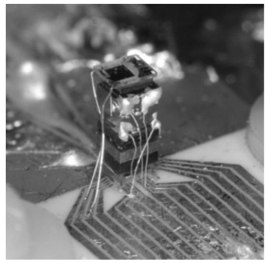


Solar day: interval between two transits of the sun.



Measurement methods estimate the solar day lengths e.g. water clocks (-270 JC), atomic clocks (half of 20th).





period	technology	accuracy
-270	water clock	Na
1335	mechanical	10^{-1} s
1928	quartz	10^{-10} s
1946	atomic clock	10^{-18} s

Time subdivision:

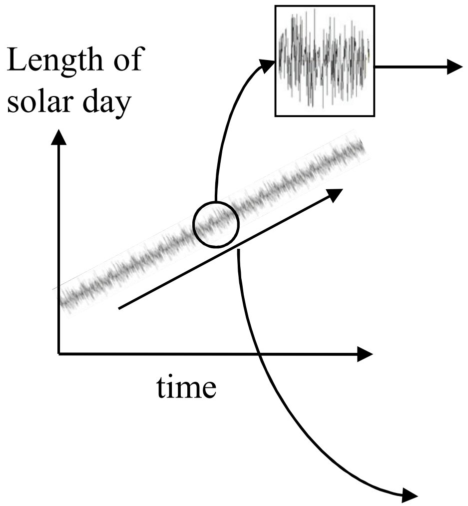
The Egyptians subdivided daytime and nighttime into twelve hours each since at least 2000 BC.

In 1267, the medieval scientist Roger Bacon stated the times as a number of hours, minutes and seconds.

$$\text{solar second} = \frac{\textit{length of solarday}}{86400}$$

(i.e. $24 \times 60 \times 60$)

Clock synchronization
"Coordinated Universal Time (UTC)" (2)



The length of solar days changes over the year (+/- 25 s)

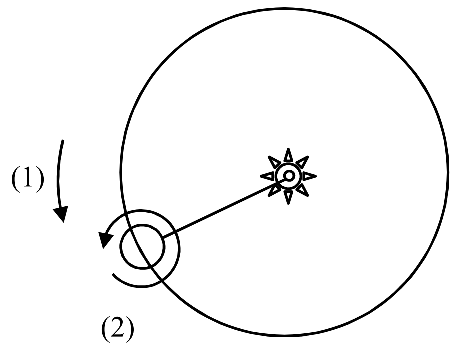
The mean solar day is $= \sum_{i=1}^{n} \frac{solarday_i}{n}$

The mean solar second is $= \sum_{i=1}^{n} \frac{solarday_i}{n \times 86400}$

In 1940's it was established that the period of the earth's rotation slows down. This results in two main consequences:

(1) the number of days per year reduce (-300 Millions of years → today : -35 days).

(2) the day become longer all the time (1820 → 2010: + 2s).



Clock synchronization
"Coordinated Universal Time (UTC)" (3)

In 1948 has been introduced the Atomic Clock, based on Cesium 133, to compute transits of the sun.

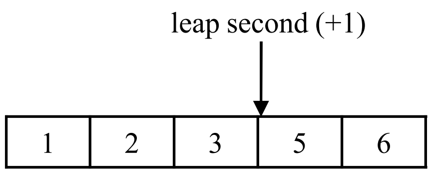
9 192 631 770 transitions of Cesium 133 equals 1 solar second at year 1948.

International Atomic Time (TAI) is the mean of Atomic Clocks since 1st January 1958 (beginning of logical time).

Time offset Δt with the TAI reference (1948)

	TAI Time offset Δt	
	1948	**2010**
1 second	0	+3 10^{-8} TAI seconds
1 day	0	+0,003 TAI seconds
1 year	0	+1,09 TAI seconds

Since 1948, to adjust the TAI time to the solar one we introduce leap seconds when solar time – TAI time > 0,8 TAI second:



We have introduced around 37 leap seconds since 1958.

TAI with leap seconds is the **Coordinated Universal Time (UTC)**

Diffusion of UTC

Mode	**Precision**
Radio	+/- 10ms
Earth satellite	+/- 0.5ms

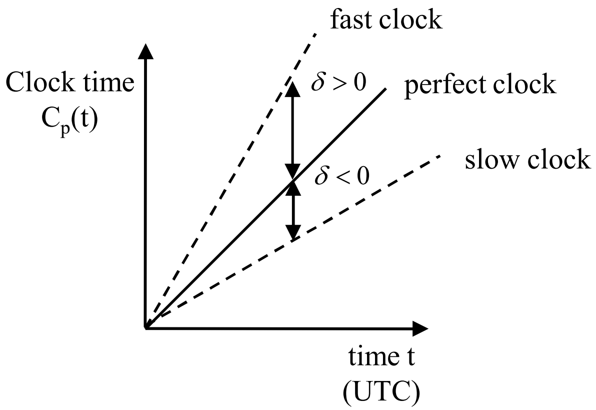
Clock synchronization
“Coordinated Universal Time (UTC)” (4)

Clock skewing denotes the extend to which the frequency differs from UTC.

Time of a clock: the time of a clock in a machine p is given by the function $C_p(t)$ where $C_p(t) = t$ for all t is a perfect clock.

Offset clock: is the difference between the time reported by a clock and UTC. The offset at time t of a clock p is $\delta = C_p(t) - t$.

Frequency: is the rate at which a clock progresses. The frequency at time t of a clock p is $C'_p(t) = dC_p(t)/dt$.

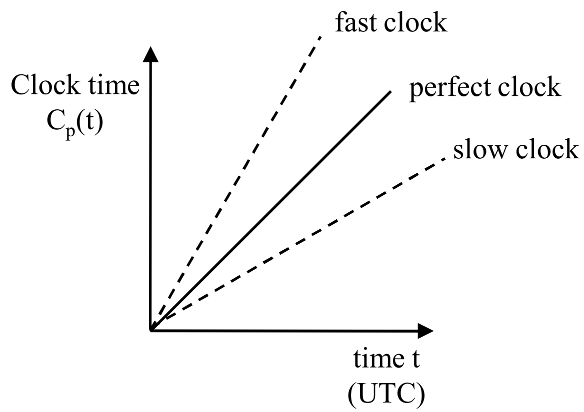


	$\frac{dC_p(t)}{dt}$
Fast clock	>1
Perfect clock	=1
Slow clock	<1

Clock synchronization
"Coordinated Universal Time (UTC)" (5)

Clock skewing denotes the extend to which the frequency differs from UTC.

Skew: the skew of a clock is the difference between the frequency of the clock and the perfect clock.
The skew at time t of a clock p is $\left(C_p(t) - t\right)' = C_p'(t) - 1 = \dfrac{dC_p(t)}{dt} - 1$.

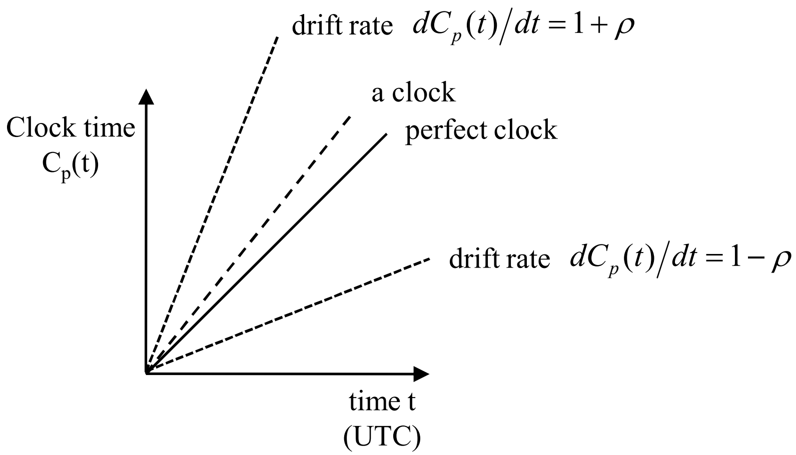


	$\frac{dC_p(t)}{dt}$	skew
Fast clock	>1	$C_p'(t) - 1 > 0$
Perfect clock	=1	$C_p'(t) - 1 = 0$
Slow clock	<1	$C_p'(t) - 1 < 0$

Clock synchronization
"Coordinated Universal Time (UTC)" (6)

Clock skewing denotes the extend to which the frequency differs from UTC.

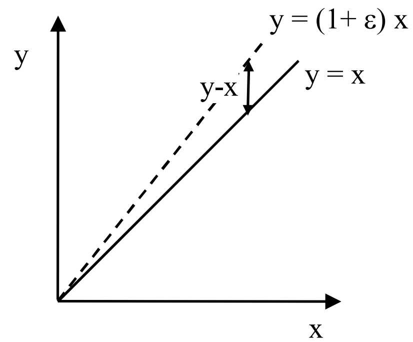
Maximum drift rate: if a skew is bounded by ρ, clock diverge at rate (frequency) in the range $1-\rho \le \frac{dC_p(t)}{dt} \le 1+\rho$,
ρ is specified by the manufacturer and is known as the maximum drift rate.



Clock synchronization
"Coordinated Universal Time (UTC)" (7)

Clock skewing denotes the extend to which the frequency differs from UTC.

The relationship with the mathematical analysis: is obtained with a reformulation as a linear function y=ax with a the tangent of the function noted as 1+ ε.



$y = (1+\varepsilon)\,x$	is the time of the clock $C_p(t)$
y-x	is the offset δ
$1+\varepsilon$	is the frequency $dC_p(t) / dt$
ε	is the skew $dC_p(t) / dt - 1$
$\lvert\varepsilon\rvert < \varepsilon_{max}$	ε_{max} is maximum drift rate ρ

Clock synchronization
"Coordinated Universal Time (UTC)" (8)

Clock skewing denotes the extend to which the frequency differs from UTC.

e.g. a computer synchronizes with an UTC server to estimate the clock skew, the synchronization is done one time a day and the first day the clock is adjusted with UTC then $\delta = 0$.

day	**0**	**1**	**2**	**3**
Cp(t) "h:m:s:ms"	17:5:06:0	13:10:50:0	10:58:15:0	12:24:58:0
Cp(t) "s"	0	72344	150789	242392
t (UTC) "h:m:s:ms"	17:5:06:0	13:10:49:241	10:58:13:355	12:24:55:323
t (UTC) "s"	0	72343,241	150787,355	242389,323
δ "ms"	0	759	1645	2677
Skew		1,04917E-05	1,09094E-05	1,10442E-05

$$\delta = C_p(t) - t$$

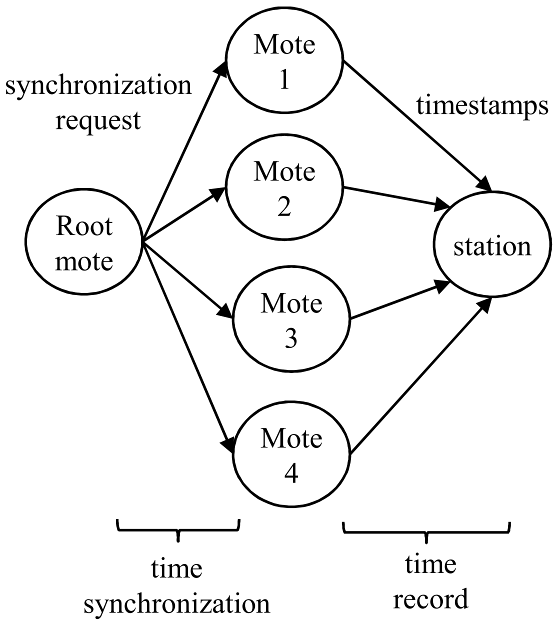
$$= C_p'(t) - 1 = \frac{dC_p(t)}{dt} - 1$$

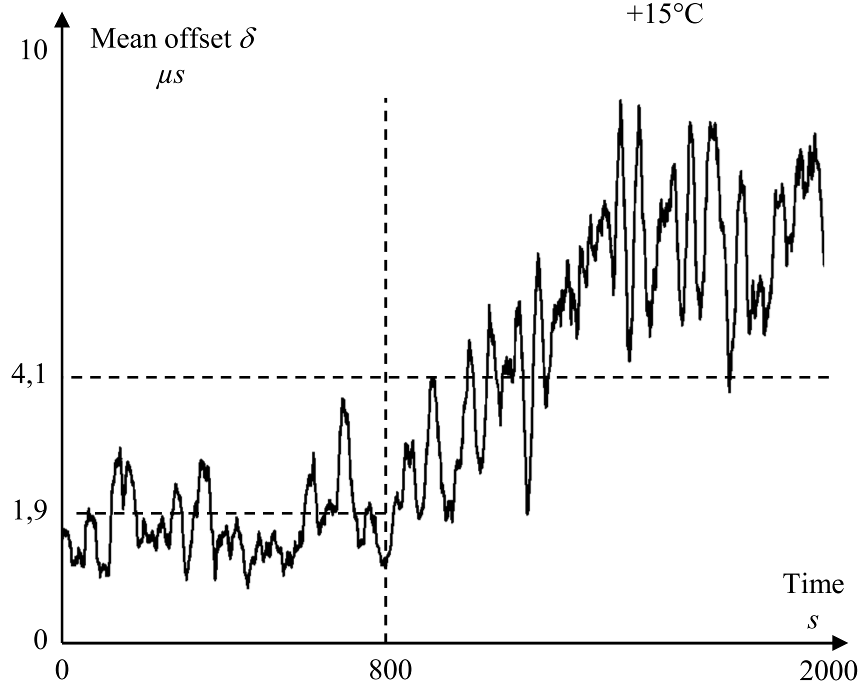
$$e.g. \quad 1{,}0491 \times 10^{-5} = \frac{72344}{72343{,}241} - 1$$

Clock synchronization
"Coordinated Universal Time (UTC)" (9)

Clock skewing denotes the extend to which the frequency differs from UTC.

e.g. a Wireless Sensor Network (WSN) with one root and four standard motes with a base station. The motes have a synchronization period of 5 s. The base station collects the mote time every second. At 800 s, a temperature variation of 15°C is provoked in one of the motes, decreasing the performance of 215.78% [J.M. Castillo-Secilla, Electronics Letters, 2013].



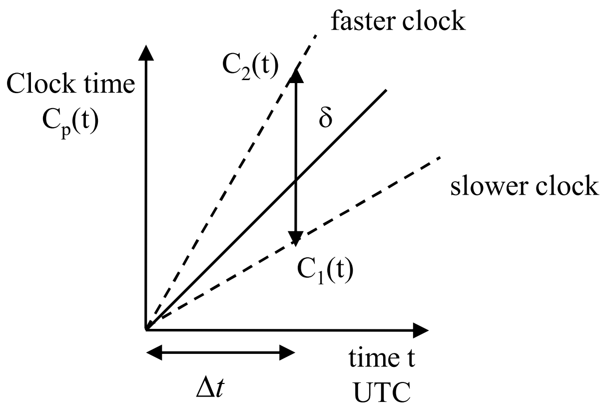


Clock synchronization
"Coordinated Universal Time (UTC)" (10)

Clock skewing denotes the extend to which the frequency differs from UTC.

Maximum drift gap: if two clocks are drifting from UTC in the opposite direction, after a time Δt they were synchronized, they may be as much as $2 \times \rho \times \Delta t$ considering a same maximum drift rate.

To bound the skew to an offset δ, we must synchronize every $\frac{\delta}{2\rho}$



$$C_2(\Delta t) - C_1(\Delta t) = \delta = (1+\rho)\Delta t - (1-\rho)\Delta t$$

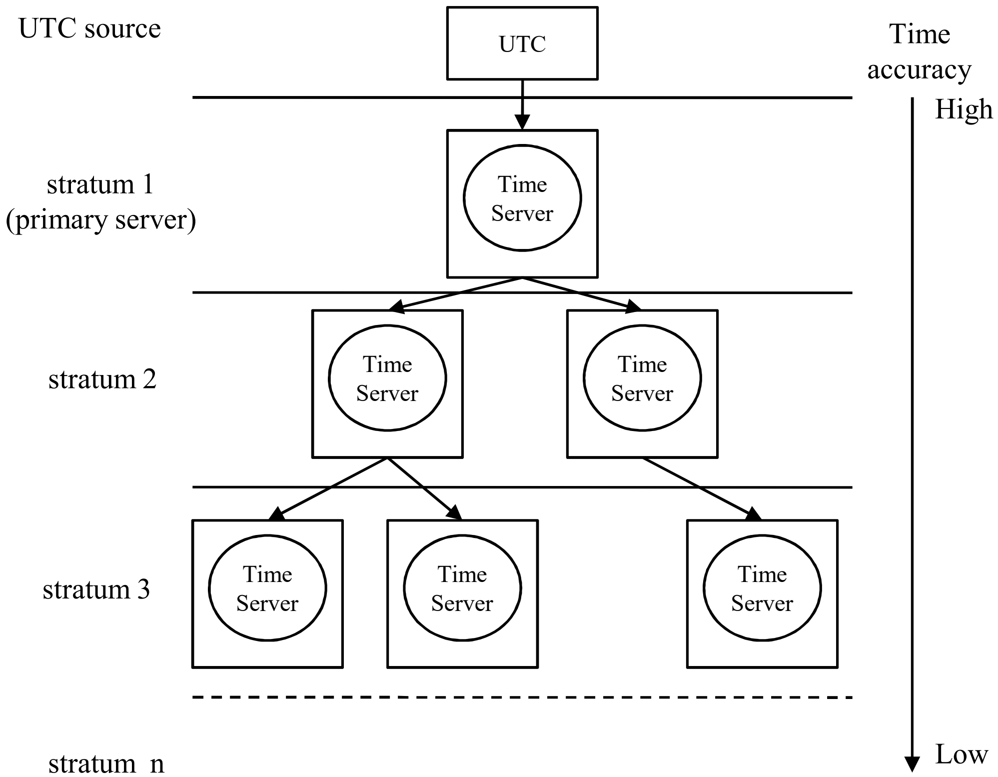
$$\Delta t = \frac{\delta}{2\rho}$$

Clock synchronization

Methods	**Reference Time**	**Server**	**Network**	**Precision**
Network Time Protocol	UTC	passive	Internet	estimation message delay
Global Positioning System (GPS)			UHF	clock offset estimation
Berkeley Algorithm	mean time	active	Ethernet	none
Reference Broadcast Synchronization (RBS)	clock offsets		Wireless	clock skew

Clock synchronization
"Network Time Protocol (NTP)" (1)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

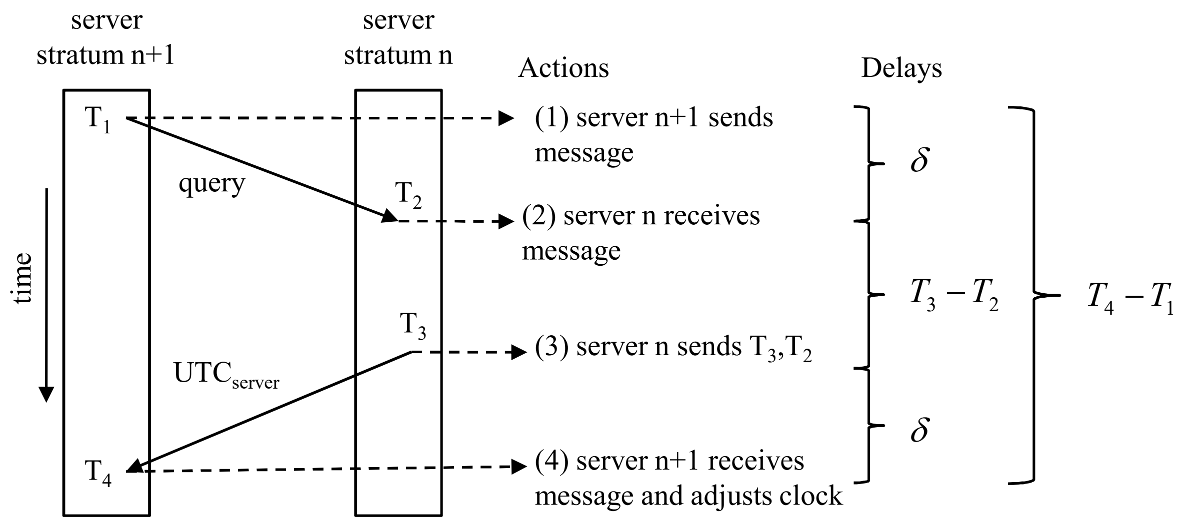


Clock synchronization
"Network Time Protocol (NTP)" (2)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

The NTP supports pair-wise exchange, a pair of servers exchanges message bearing timestamps information.

The problem is that when contacting a server, message delay will have outdated the reported time.



δ is the estimation for the delay of exchange. We assume that the propagation delay between the server is "roughly" the same.

$$\delta = \frac{(T_2 - T_1) + (T_4 - T_3)}{2}$$

Applying local comparison (T4 against T1, T3 against T2) makes sense, δ is then reformulated.

$$\delta = \frac{(T_4 - T_1) - (T_3 - T_2)}{2}$$

Clock synchronization
"Network Time Protocol (NTP)" (3)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

At (4) synchronization is done with: θ is the clock offset between the servers n and n+1

$$\theta = C_n(t) - C_{n+1}(t) \quad \begin{cases} C_n(t) = T_3 + \delta \\ C_{n+1}(t) = T_4 \\ \theta = T_3 + \delta - T_4 \end{cases}$$

Clock synchronization
"Network Time Protocol (NTP)" (4)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

At (4) synchronization is done with: θ is the clock offset between the servers n and n+1

delay of time

(1) if $\theta \geq 0$ *then* $C_n(t) \geq C_{n+1}(t)$

$C_{n+1}(t)$ is a slow clock

we apply $C_{n+1}(t) = C_{n+1}(t) + \theta$

Clock synchronization
“Network Time Protocol (NTP)” (5)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

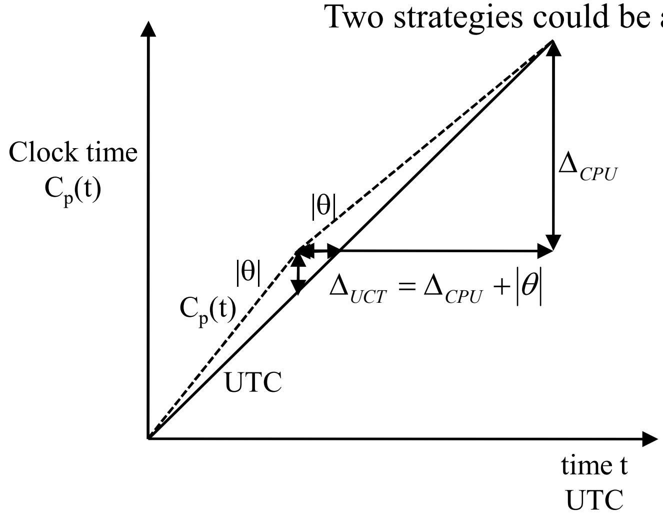
At (4) synchronization is done with: θ is the clock offset between the servers n and n+1

ahead of time

(2) if $\theta \leq 0$ *then* $C_n(t) \leq C_{n+1}(t)$

$C_{n+1}(t)$ is a fast clock

Computer cannot backward in time, off course. Then we reduce the speed of the server n+1 until θ equals 0.
Two strategies could be applied



First strategy, we fix Δ_{CPU} to get a speedup ratio $r = \in]0,1[$ to be applied to the CPU Frequency F_{NORM}

$$F_{RED} = r \times F_{NORM}$$

$$r = \frac{F_{RED}}{F_{NORM}} = \frac{\Delta_{CPU}}{\Delta_{UCT}} = \frac{\Delta_{CPU}}{\Delta_{CPU} + |\theta|}$$

Clock synchronization
“Network Time Protocol (NTP)” (6)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

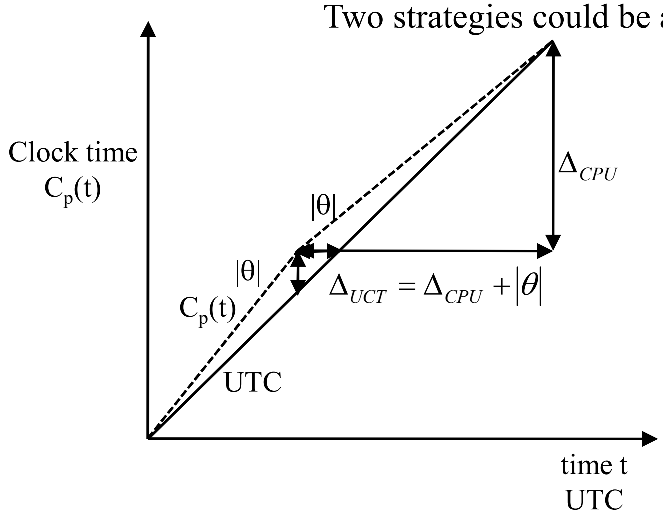
At (4) synchronization is done with: θ is the clock offset between the servers n and n+1

ahead of time

(2) if $\theta \leq 0$ *then* $C_n(t) \leq C_{n+1}(t)$

$C_{n+1}(t)$ is a fast clock

Computer cannot backward in time, off course. Then we reduce the speed of the server n+1 until θ equals 0.
Two strategies could be applied



Second strategy, we fix the speedup ratio $r = \in]0,1[$ to get Δ_{CPU}

$$\Delta_{CPU} = \frac{r}{1-r}|\theta|$$

Clock synchronization
"Network Time Protocol (NTP)" (7)

Network Time Protocol (NTP) defines an architecture for a time service and a protocol to distribute time information over the Internet. Time is provided by a network of servers located across the internet, organized as a tree. Servers of higher stratums synchronize time from the lowest ones.

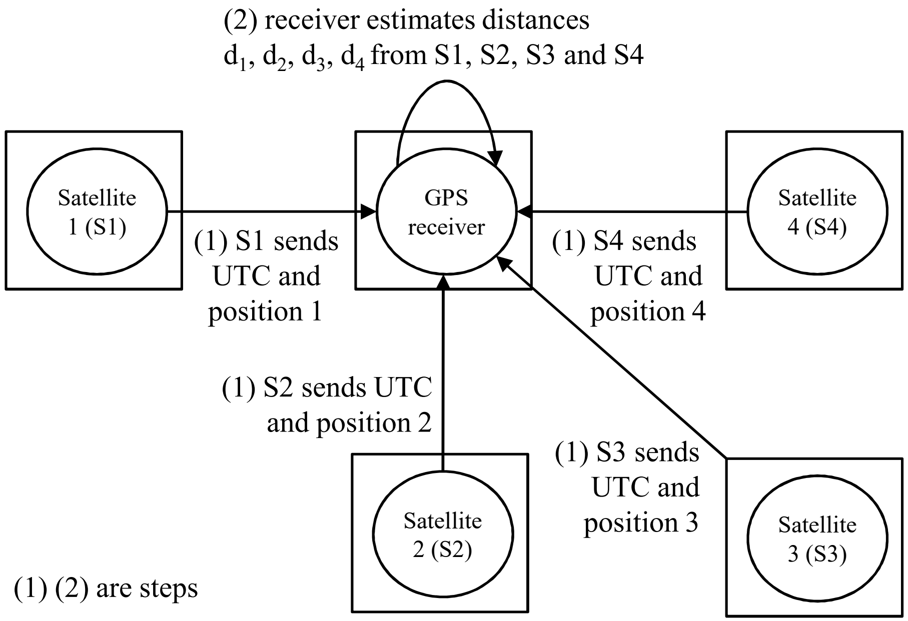
e.g.	The interruption service at the client level (stratum n+1), T4 – T1, is	710 ms	
	The interruption service at the server level (stratum n), T3 – T2, is	550 ms	
	The propagation delay δ is then	80 ms	$\delta = \frac{(T_4 - T_1) - (T_3 - T_2)}{2}$
	Considering a server / client clock gap T3 – T4 = –264 ms, the server / client clock offset is	-184 ms	$\theta = T_3 + \delta - T_4$
	The client has a fast clock with $\theta < 0$, we want to fix the synchronization at a CPU frequency rate r > 0,95, Δ_{CPU} is then	3496 ms	$\Delta_{CPU} = \frac{r}{1-r}\lvert\theta\rvert = 3496\ ms$
	We can fix Δ_{UCT} the real-time synchronization	3680 ms	$\Delta_{UCT} = \Delta_{CPU} + \lvert\theta\rvert = 3496 + 184 = 3680\ ms$

Clock synchronization

<table>
<tr><th>Methods</th><th>Reference Time</th><th>Server</th><th>Network</th><th>Precision</th></tr>
<tr><td>Network Time Protocol</td><td rowspan="2">UTC</td><td rowspan="2">passive</td><td>Internet</td><td>estimation message delay</td></tr>
<tr><td>Global Positioning System (GPS)</td><td>UHF</td><td>clock offset estimation</td></tr>
<tr><td>Berkeley Algorithm</td><td>mean time</td><td rowspan="2">active</td><td>Ethernet</td><td>none</td></tr>
<tr><td>Reference Broadcast Synchronization (RBS)</td><td>clock offsets</td><td>Wireless</td><td>clock skew</td></tr>
</table>

Clock synchronization
"Global Positioning System (GPS)" (1)

Global Positioning System (GPS) is a satellite based distributed system:
- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

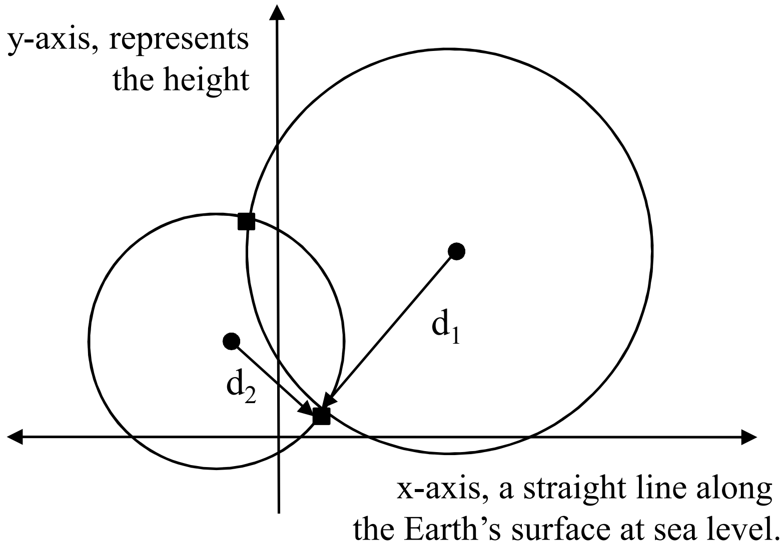


Clock synchronization
"Global Positioning System (GPS)" (2)

Global Positioning System (GPS) is a satellite based distributed system:

- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

e.g. in two dimensions



● is a satellite position

■ a point at some given distances from the two satellites, the highest point is ignored (it is located in the space)

Clock synchronization
"Global Positioning System (GPS)" (3)

Global Positioning System (GPS) is a satellite based distributed system:
- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

We assume that the timestamp from a satellite is completely accurate, when a message is received from a satellite i we have:

$$\Delta_i = (T_r + \Delta_r) - T_i = (T_r - T_i) + \Delta_r$$

where,
Δ_i is the measured delay by the receiver r from satellite i,
T_i is timestamp of satellite i,
T_r is synchronized time of the GPS receiver,
Δ_r is the deviation of the receiver (offset),

Clock synchronization
"Global Positioning System (GPS)" (4)

Global Positioning System (GPS) is a satellite based distributed system:

- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

As signals travel with the speed of the light c, we have:

$$d_i^m = c\Delta_i = c\big((T_r - T_i) + \Delta_r\big)$$

$$d_i^m = d_i^r + \varepsilon$$

$$d_i^r = c(T_r - T_i)$$

$$\varepsilon = c\Delta_r$$

where,
c is the speed of the light (299 792 458 m.s^{-1}),
d_i^m is the measured (m) distance from the satellite i,
d_i^r is the real (r) distance from the satellite i,
ε is the distance error resulting of the clock deviation,

Clock synchronization “Global Positioning System (GPS)” (5)

Global Positioning System (GPS) is a satellite based distributed system:
- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

The real distance is simply computed as

$$d_i^m = d_i^r + \varepsilon$$

$$d_i^r = \sqrt{(x_i - x_r)^2 + (y_i - y_r)^2 + (z_i - z_r)^2}$$

$$\varepsilon = c\Delta_r$$

where,
(x_i, y_i, z_i) denote the coordinates of satellite i,
(x_r, y_r, z_r) denote the coordinates of the receiver,

Finding the position corresponds to solving the system of linear equation to obtain x_r, y_r, z_r and Δ_r using four satellites (at least).

This belongs to the numerical analysis field, using algorithms such as the matrix decomposition or iterative methods.

Thus, a GPS measurement will also give an account of the of the actual time by approximating Δ_r.

Clock synchronization
“Global Positioning System (GPS)” (6)

Global Positioning System (GPS) is a satellite based distributed system:
- GPS uses 29 satellites each circulating in an orbit at a height of approximately 20 000 km,
- each satellite has up to four atomic clocks, regularly calibrated from Earth,
- a satellite continually broadcasts its position, and time stamps in each message with its local time,
- this broadcasting allows every receiver on Earth to compute its own position using at least four satellites.

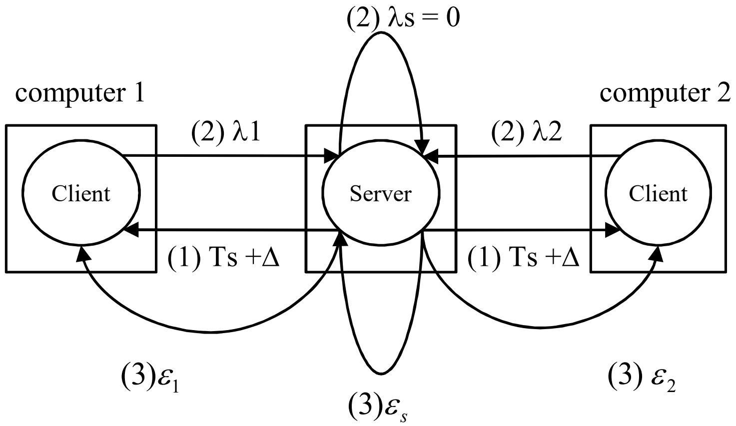
Misc issues, so far we have assumed that measurement are perfectly accurate, they are not:
- GPS does not take into account the leap seconds (deviation since 1st January 2006),
- atomic clocks in the satellites are not perfect in synch,
- the position of a satellite is not known precisely,
- the receiver’s clock has a finite accuracy,
- the signal propagation speed is not a constant,
- time dilatation affects the clocks,
- etc.

Clock synchronization

Methods	Reference Time	Server	Network	Precision
Network Time Protocol	UTC	passive	Internet	estimation message delay
Global Positioning System (GPS)			UHF	clock offset estimation
Berkeley Algorithm	mean time	active	Ethernet	none
Reference Broadcast Synchronization (RBS)	clock offsets		Wireless	clock skew

Clock synchronization
"Berkeley Algorithm"

Berkeley Algorithm: this method is suitable for system in which no machine has a time signal. Time server is active, pooling every computers of the network periodically. Based on their answers, the time server computes mean drift and tell to machine how to set up their clocks. It works in three steps (1) (2) (3).



		(1) Clock	(2) $\lambda i = Ti - Ts$	(3) $\varepsilon_i = \bar{\lambda} - \lambda_i$	$T_i + \varepsilon_i$
Ts		3:00	0	+5	3:05
Ti	T1	3:25	+25	-20	3:05
	T2	2:50	-10	+15	3:05

$$\bar{\lambda} = \sum_{i=1}^{n} \frac{\lambda_i}{n} = +5$$

with

- (1)(2)(3) — are events
- Ti (i =s,1,2) — are computer times
- λi — clock drift between a computer i and server (λi = Ti-Ts)
- Δ — time to transmit a message with $\Delta_{min} \leq \Delta \leq \Delta_{max}$
- $\bar{\lambda}$ — mean clock drift

The communication delay is bounded in the case of a local network: $\Delta_{min} \leq \Delta \leq \Delta_{max}$

Due to communication an error $|\varepsilon| < \frac{\Delta_{max} - \Delta_{min}}{2}$ is introduced at (1), when the server sends the clock value.

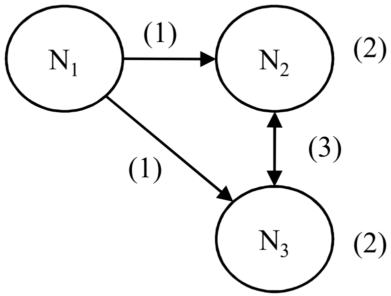
Rest of exchanges (2) (3) are computed from some delta parameters, these steps have no impact on precision.

Clock synchronization

<table>
<tr><th>Methods</th><th>Reference Time</th><th>Server</th><th>Network</th><th>Precision</th></tr>
<tr><td>Network Time Protocol</td><td rowspan="2">UTC</td><td rowspan="2">passive</td><td>Internet</td><td>estimation message delay</td></tr>
<tr><td>Global Positioning System (GPS)</td><td>UHF</td><td>clock offset estimation</td></tr>
<tr><td>Berkeley Algorithm</td><td>mean time</td><td rowspan="2">active</td><td>Ethernet</td><td>none</td></tr>
<tr><td>Reference Broadcast Synchronization (RBS)</td><td>clock offsets</td><td>Wireless</td><td>clock skew</td></tr>
</table>

Clock synchronization
"Reference Broadcast Synchronization (RBS)" (1)

Reference Broadcast Synchronization (RBS) is dedicated to sensor network where no time server is available. The communication between nodes must be restricted to save energy. The RBS algorithm looks-like the Berkeley Algorithm, it deviates of the two-ways protocol (i.e. keep receiver and sender into synch) by letting only receivers synchronize.



(1) A sender broadcasts a reference message *m*.

(2) A receiver *p* simply records the local time $T_{p,m}$ that it received with *m*.

(3) Two nodes *p,q* can exchange each other's delivery times in order to estimate their relative offset. They store this offset, there is no need to adjust the clocks.

$$Offset[p,q] = \frac{1}{M}\sum_{k=1}^{M} T_{p,k} - T_{q,k}$$

The mean offset is computed with a geometric mean, other metrics could be applied.

Clock synchronization
"Reference Broadcast Synchronization (RBS)" (2)

Reference Broadcast Synchronization (RBS) is dedicated to sensor network where no time server is available. The communication between nodes must be restricted to save energy. The RBS algorithm looks-like the Berkeley Algorithm, it deviates of the two-ways protocol (i.e. keep receiver and sender into synch) by letting only receivers synchronize.

e.g.

Events	**Message stacks**		
	N_1	**N_2**	**N_3**
N_1 broadcasts		N_{11},10	N_{11},12
N_3 broadcasts	N_{31},14	N_{11},10 N_{31},13	N_{11},12
N_2 broadcasts	N_{31},14 N_{21},18	N_{11},10 N_{31},13	N_{11},12 N_{21},21
N_3 broadcasts	N_{31},14 N_{21},18 N_{32},24	N_{11},10 N_{31},13 N_{32},22	N_{11},12 N_{21},21
N_1, N_2 synchronize	N_{31},14 N_{21},18 N_{32},24	N_{11},10 N_{31},13 N_{32},22	N_{11},12 N_{21},21

N_{11}, N_{31}, N_{21}, N_{32} are messages

10,12,13,14, etc. are the record times $T_{p,k}$, $T_{q,k}$

$$Offset[N_1, N_2] = \frac{1}{M}\sum_{k=1}^{M} T_{p,k} - T_{q,k}$$

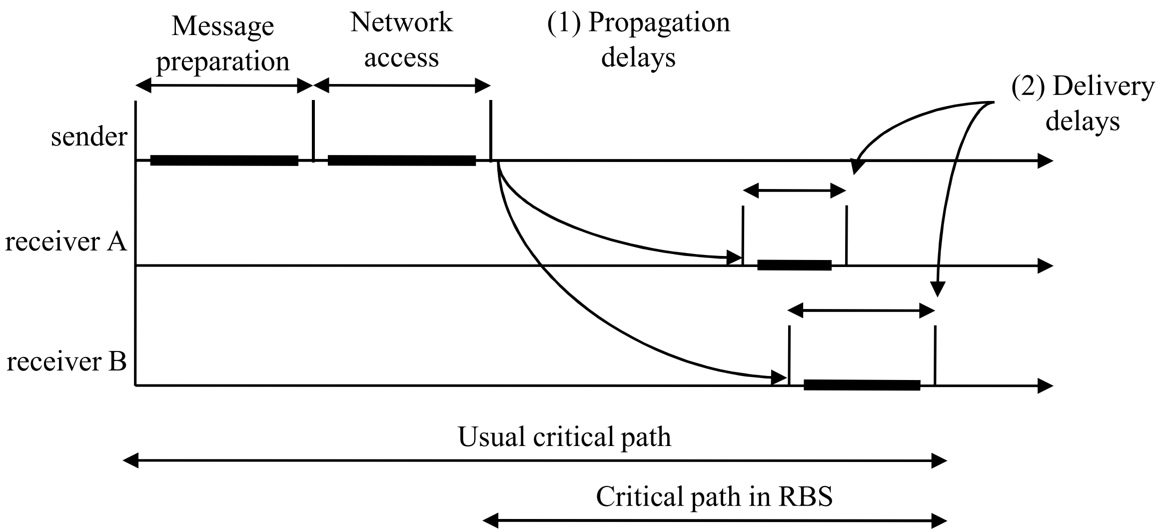
$$Offset[N_1, N_2] = \frac{1}{2}(14 - 13 + 24 - 22) = \frac{3}{2} = 1{,}5$$

where T_{11}=14, T_{12}=24, T_{21}=13, T_{22}=22

Clock synchronization
"Reference Broadcast Synchronization (RBS)" (3)

Reference Broadcast Synchronization (RBS) is dedicated to sensor network where no time server is available. The communication between nodes must be restricted to save energy. The RBS algorithm looks-like the Berkeley Algorithm, it deviates of the two-ways protocol (i.e. keep receiver and sender into synch) by letting only receivers synchronize.

Critical path: in RBS, only the receivers synchronize. As a consequence, RBS eliminates the sender-side uncertainly from the critical path, making the critical path more accurate:



(1) the propagation time in sensor network is roughly a constant.
(2) the delivery time at the receiver varies considerately less than the network access time (at the sender).