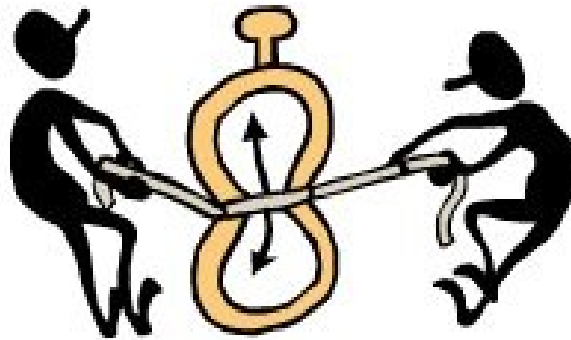


# Time Synchronization for High Latency Acoustic Networks



Affan A. Syed and John Heidemann

Presented by Daniel Sadoc Menasché



Now it's 9.00 am.

What do you mean by  
**now**?



Network

# Problem definition

- How do establish, in a distributed way, what is “now”?
- AKA **clock synchronization** (or time synchronization).
- By the way, what is time?
  - Is time a fundamental part of the structure of the universe (Newton) or is it just an intellectual structure (Kant and Leibniz)?

# Motivation

- **Military** applications: prevent enemy attacks via **alerts** when enemies are detected.
- **Environmental** applications: monitor **ecological habitats**.
- **Seismic** monitoring: a seismic signal is generated centrally, propagates and reflects. Delays are used to infer the **oil reservoir** status.

# Two Kinds of Unknowns

- Intrinsically related to the **clock**
  - **skew** and **offset**
- **Network** related
  - the **critical path** of the network

# Lies that a Clock Tell

measured  
time

real clock

perfect clock

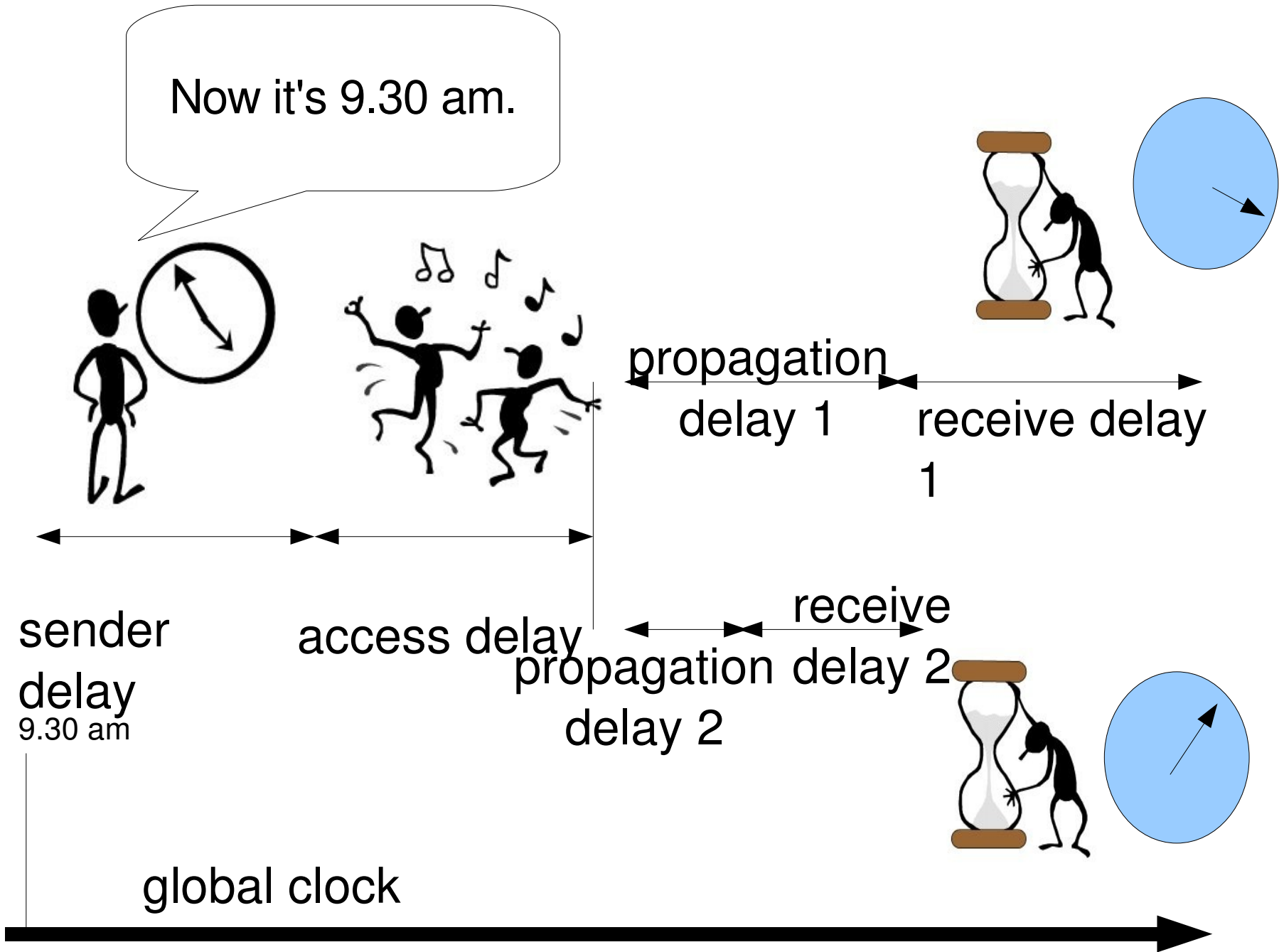
skew

offset

time



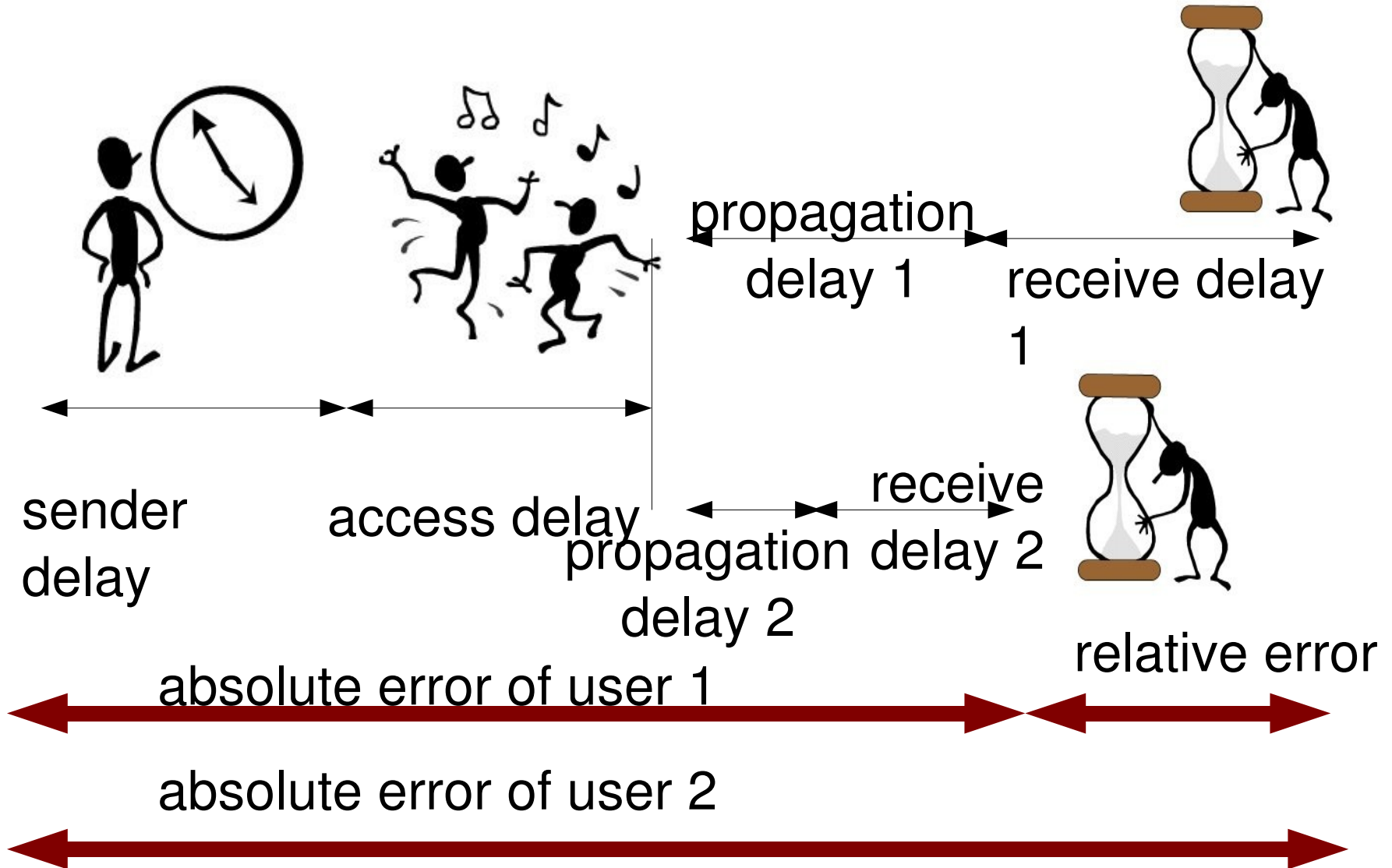
Now it's 9.30 am.



# Categories of Solutions

- There are two classical ways to solve the problem of time synchronization
  - Receiver-Receiver Synchronization (one way exchanges)
  - Sender-Receiver Synchronization (two way exchanges)
- As is, none works for underwater nets
- As we will see, the proposed method borrows ideas from both

# Receiver-Receiver Sync



Now it's 9.30 am.

propagation  
delay is  
assumed  
constant

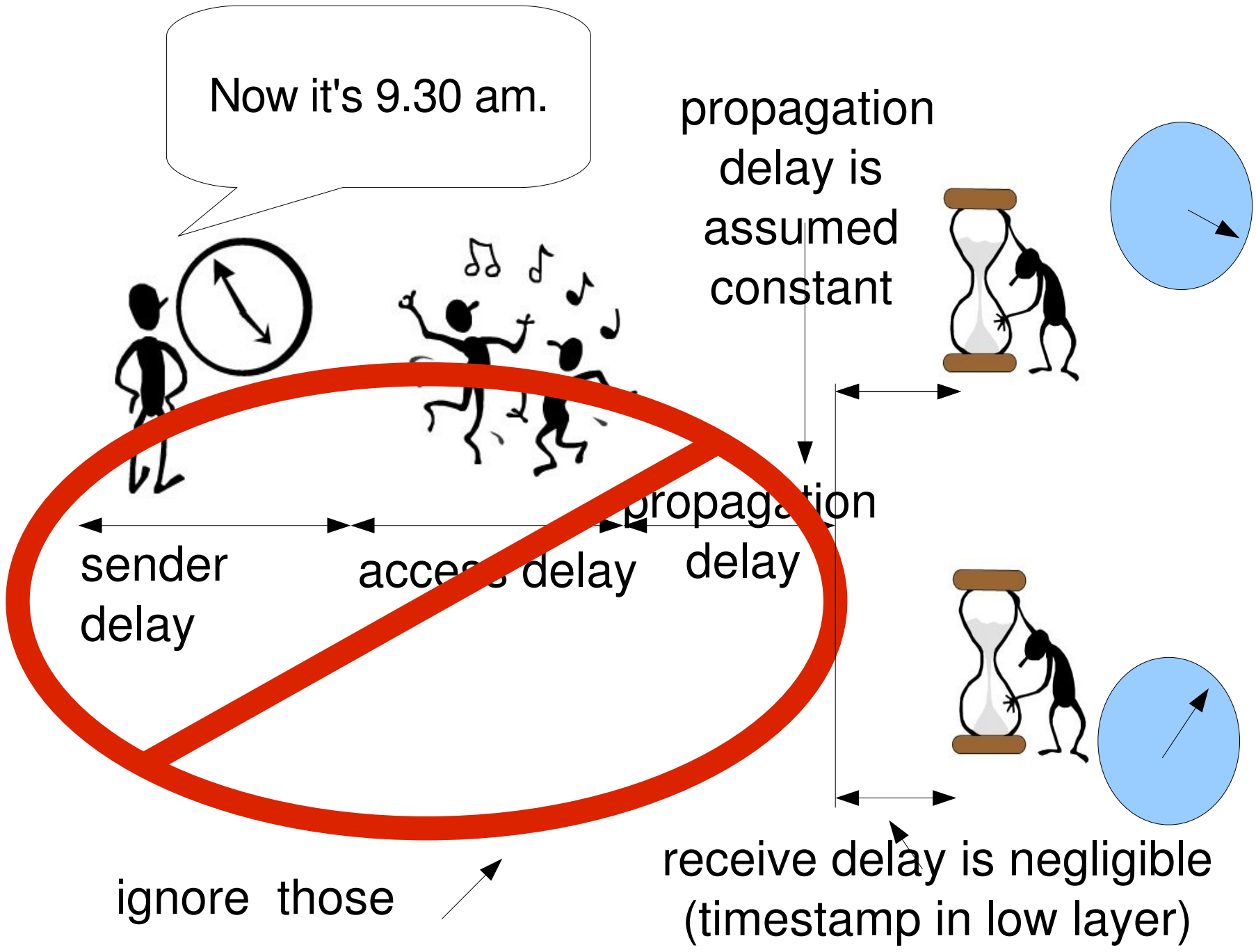
sender  
delay

access  
delay

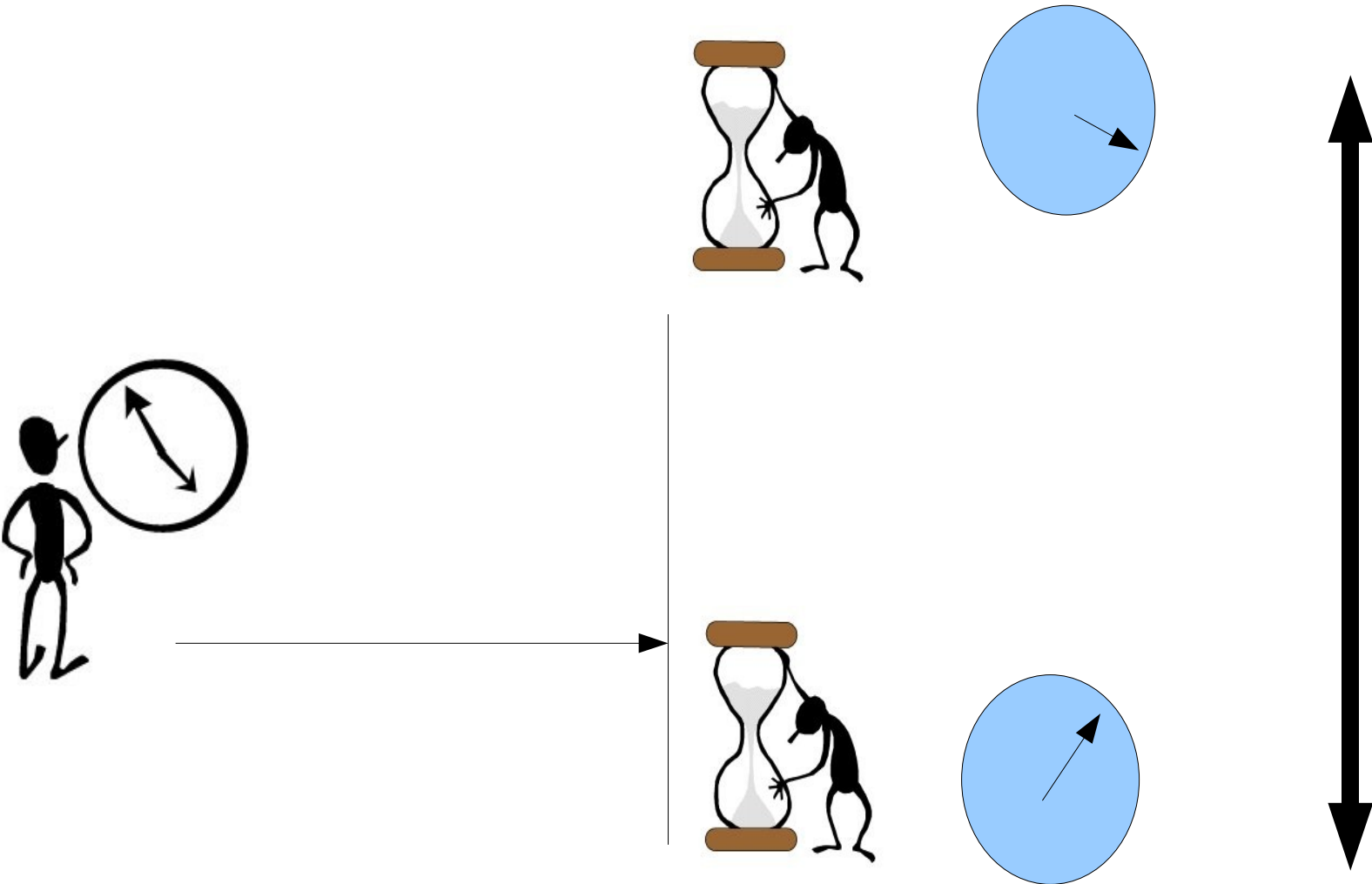
propagation  
delay

ignore those

receive delay is negligible  
(timestamp in low layer)



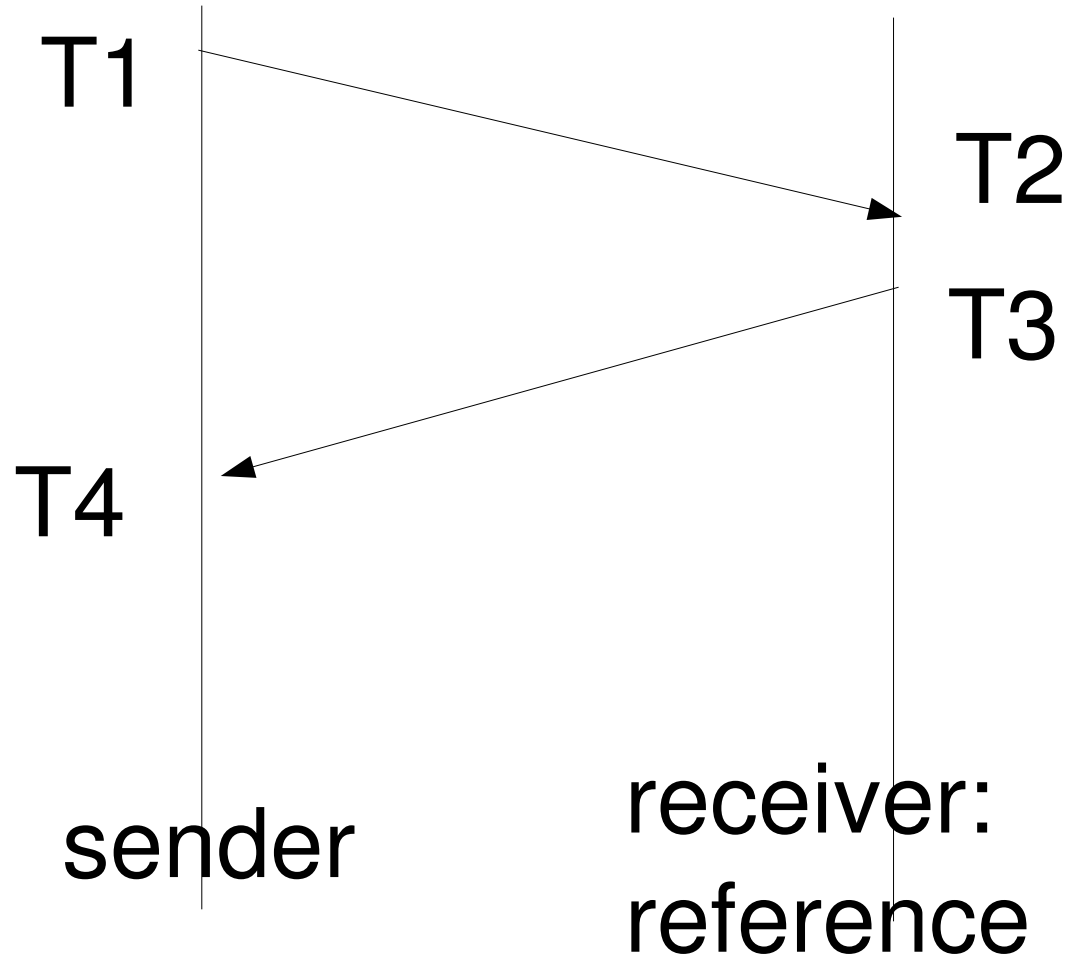
# How RBS Works



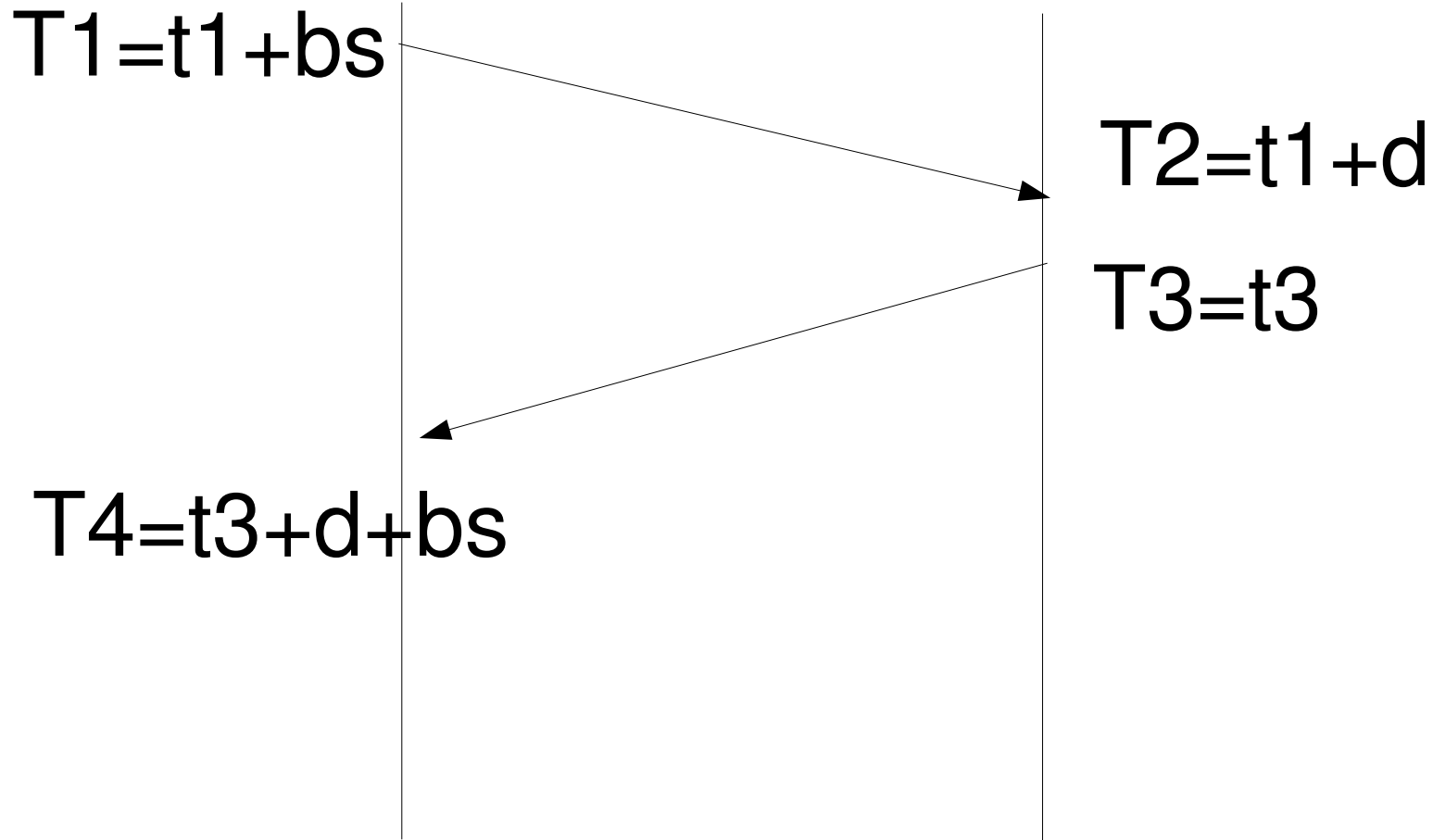
# RBS Does Not Work for Acoustic Networks

- Low data rate: 5 kbps
- Short-range: 50-500m
- **High propagation delay:  $\approx 1500$  m/s**
- Propagation delay cannot be ignored
- **Cannot** take advantage of **broadcast nature of channel**
- Cannot be used

# Sender Receiver Sync



# How TPSN Works

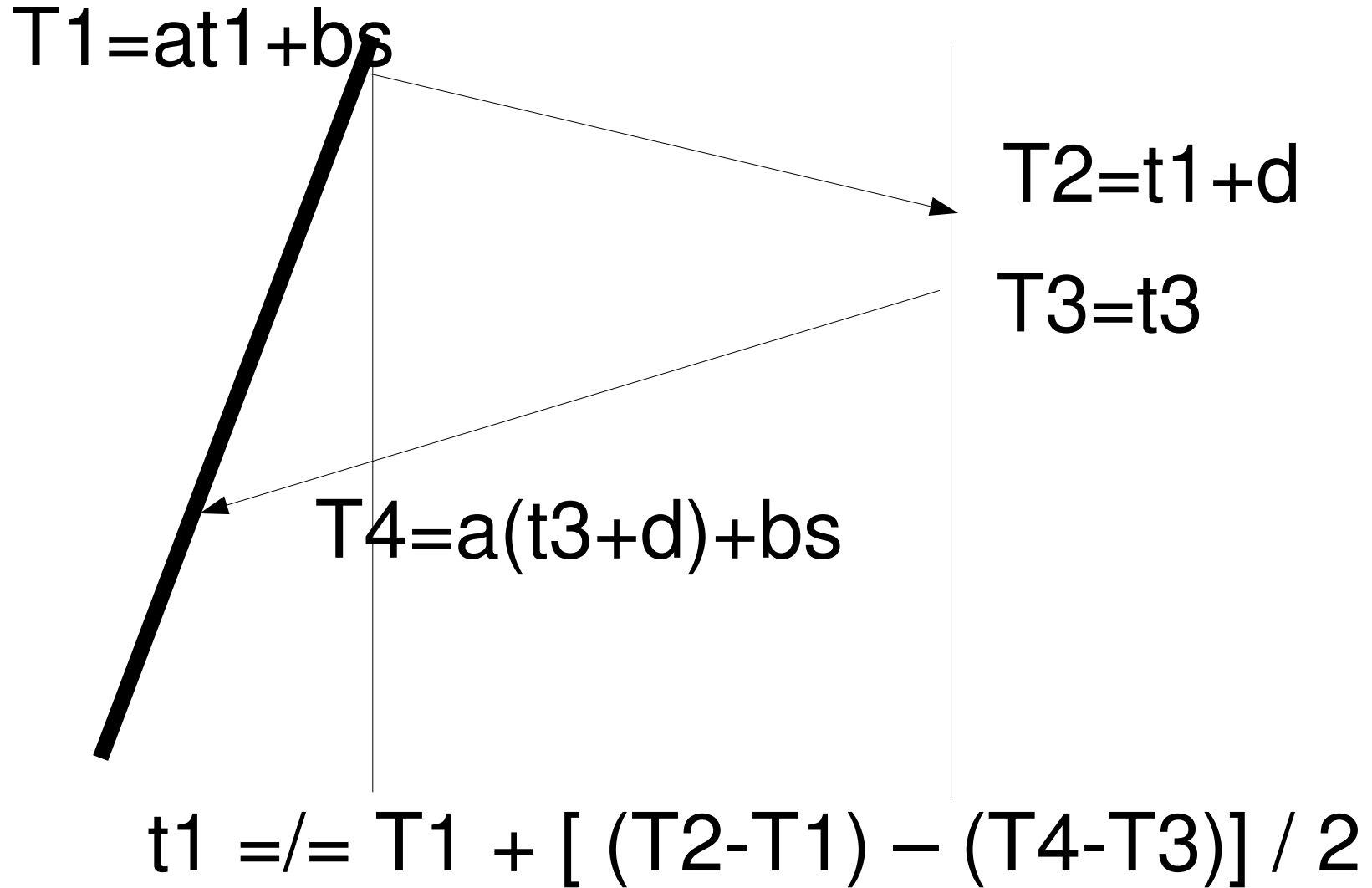


$$t1 = T1 + [ (T2-T1) - (T4-T3) ] / 2$$

# TPSN Does Not Work for Acoustic Networks

- **Low data rate: 5 kbps**
- Short-range: 50-500m
- **High propagation delay:  $\approx 1500$  m/s**
- The **skew** during the offset removal **cannot be ignored**

# TPSN Does Not Work



# To Sum Up

	RBS	TPSN
Purpose	Skew and offset removal	Offset removal
Problems with	Long delays	Skew
Category	One way exchange	Two way exchange
Good for	synchronization of receivers one to the other	synchronize sender to to receiver

# Time Synchronization for High Latency Channel (TSHL)

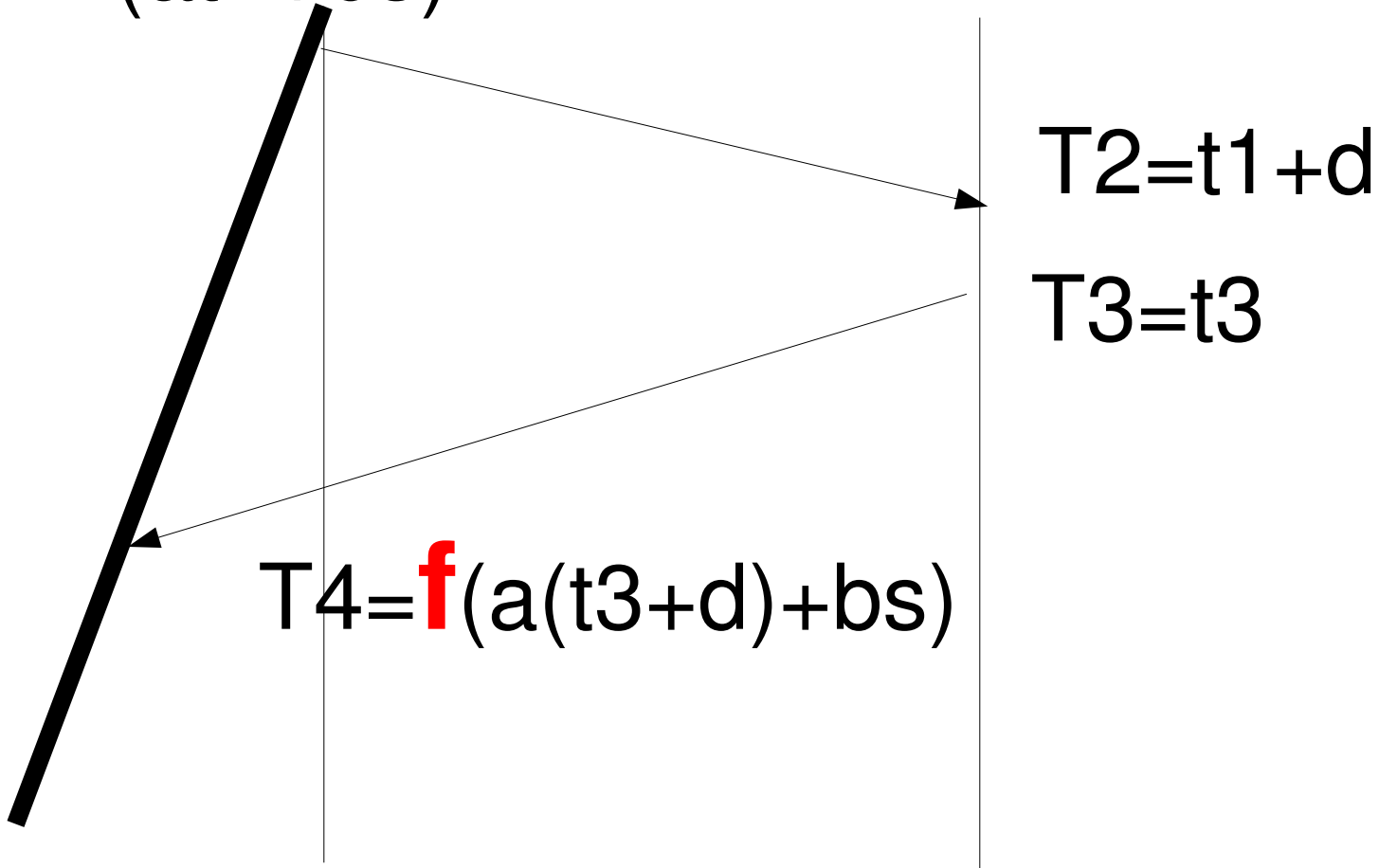
- **Skew** removal (a la RBS)
- **Offset** removal (a la TPSN)
- The **offset** removal uses the results from the **skew** removal

# Skew Removal

- Sender broadcasts multiple packets to receiver
- For each packet:
  - receiver computes **difference** between **receive time** and **send time**
- Use **linear regression** to infer skew from data
- Output: function  $f$  that removes skew

# Offset Removal

$$T1 = \mathbf{f}(at1 + bs)$$



$$T2 = t1 + d$$

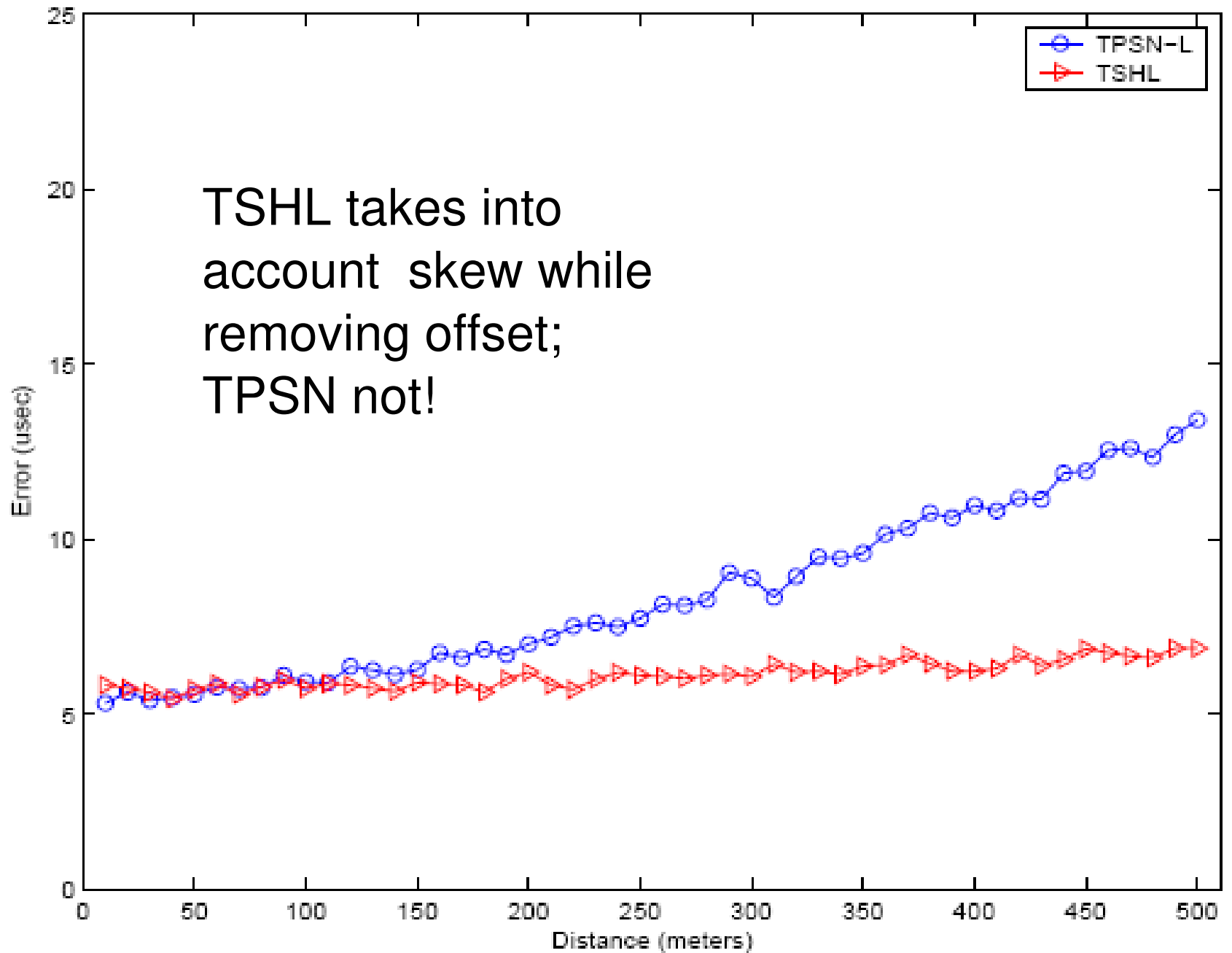
$$T3 = t3$$

$$T4 = \mathbf{f}(a(t3 + d) + bs)$$

$$t1 = T1 + [(T2 - T1) - (T4 - T3)] / 2$$

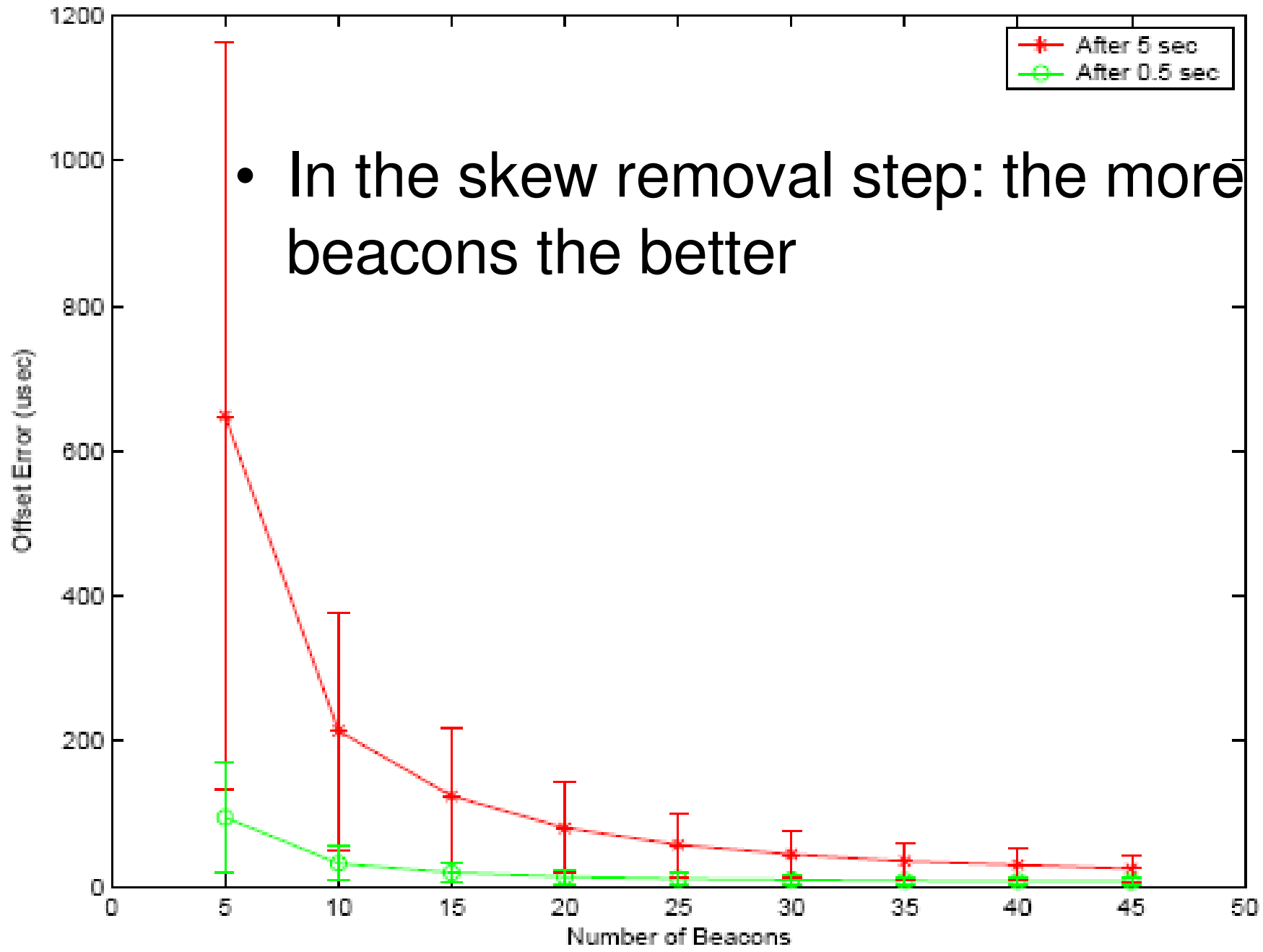
# Evaluations

- Simulation setup
  - Skew=40ppm (parts per million)
  - Offset=10 $\mu$ s
  - Number of Beacon Messages=25
  - Granularity=1 $\mu$ s
  - Receive Jitter=15  $\mu$ s

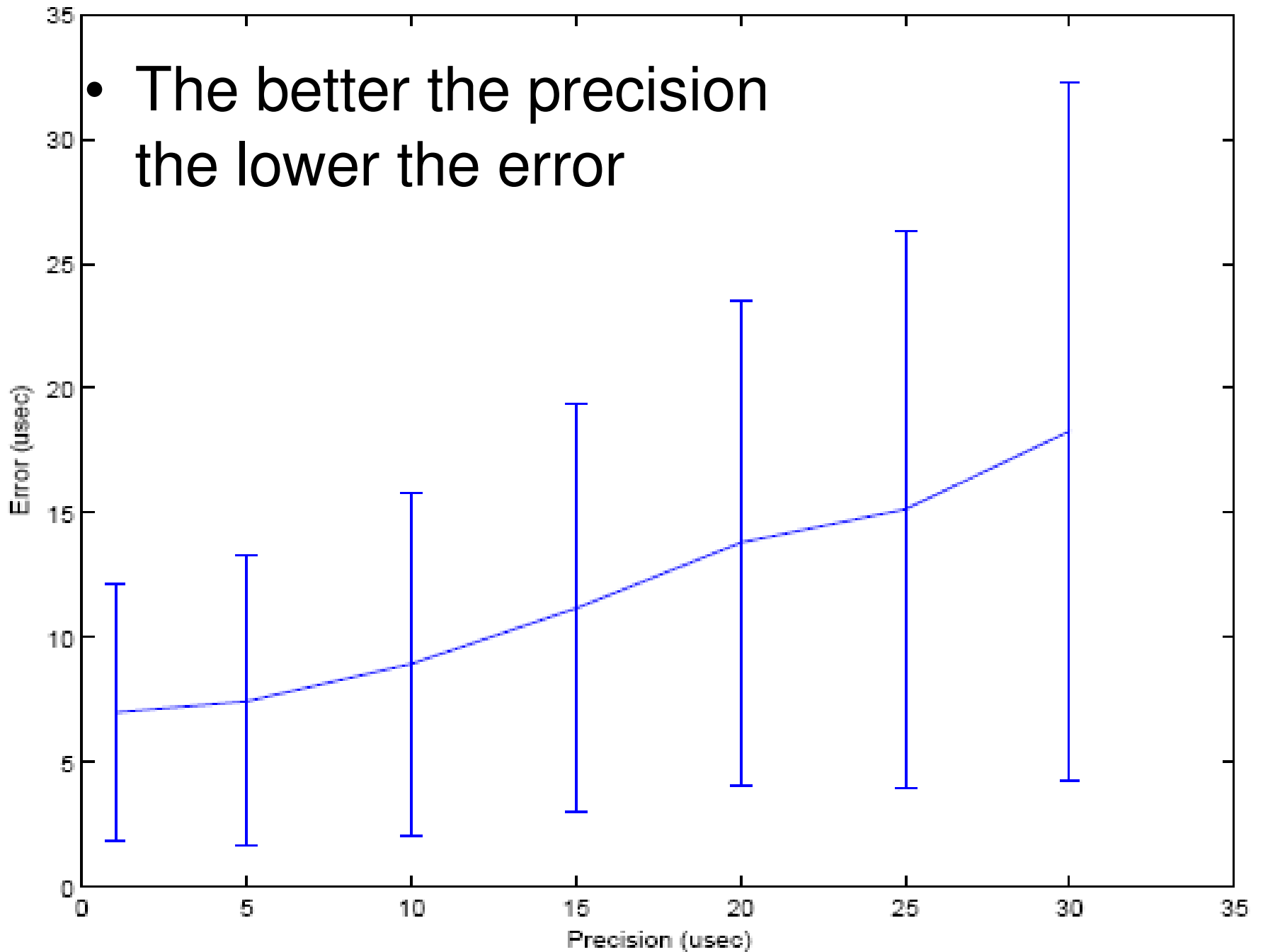


TPSN-L takes into account skew while removing offset; TSHL not!

- In the skew removal step: the more beacons the better



- The better the precision the lower the error

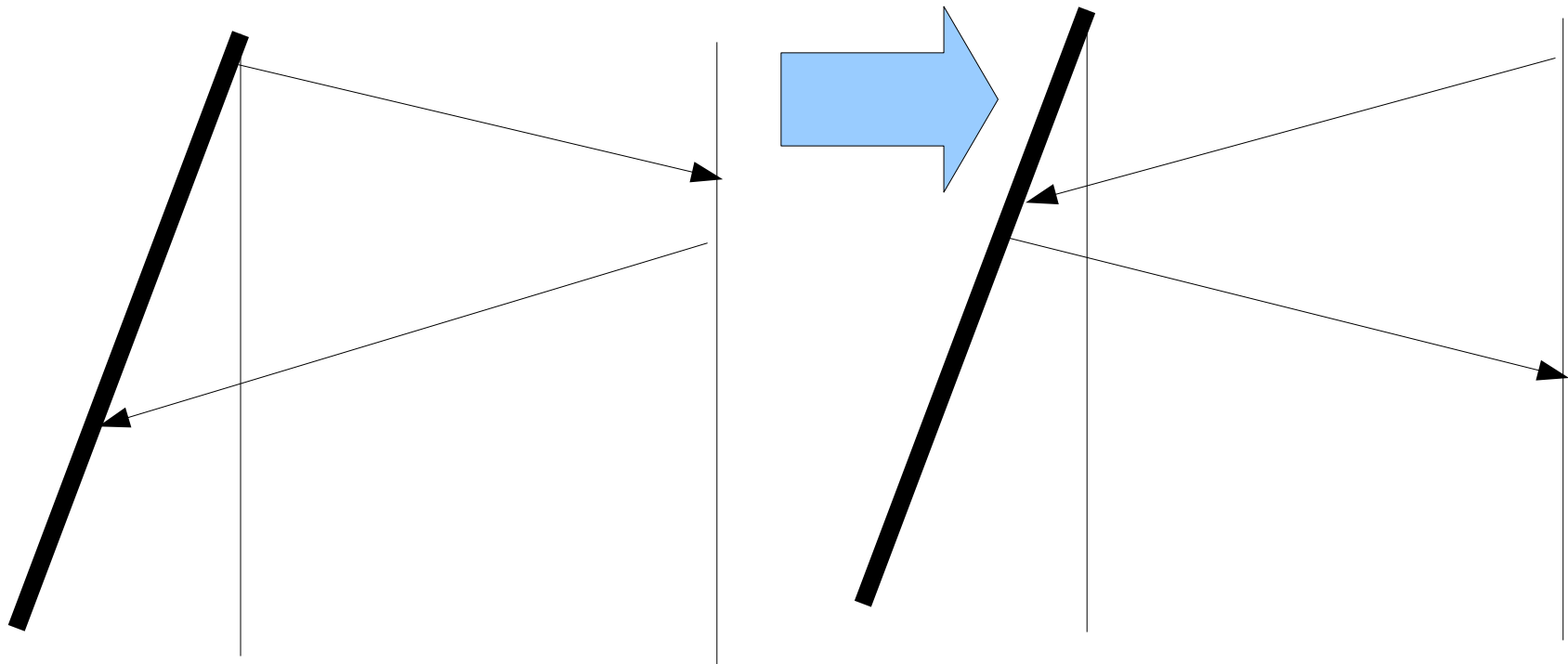


# Conclusions

- **Existing** time sync protocols for RF **perform poorly** under high propagation delays
- **Skew has to be considered even during the synchronization exchanges**

# Discussion

- Scenarios with multiple nodes were not emphasized (focus on 1 node sync)
- How frequently to sync? Application dependent
  - Broadcasting beacon messages
  - Synchronizing all the nodes (in RBS takes  $O(n^2)$ )
- Effects of contention in MAC layer



More generally: why not explicitly measuring the one way delay between the nodes?

# Backup Slides

- More slides...

# The Challenges of High Latency

$$f_S(t) = a_S t + b_S$$

$$\widehat{f}_S(t) = f_S(t) + \beta_S(t)$$

# The Challenges of High Latency (Cont'd)

- Protocols based on one-way exchanges

$$\bar{f}_B(t) = \widehat{f}_B(t) = t$$

$$\beta_R(t) = \bar{f}_B(t) - \bar{f}_R(t) = (1 - a_R)t - b_R$$

$$\beta(t + d) = \bar{f}_B(t) - \bar{f}_S(t + d) = (1 - a_S)t - (a_S d + b_S)$$

# The Challenges of High Latency (Cont'd)

- Protocols based on two-way exchange

$$T_1 = f_s(t_1), T_2 = f_B(t_1 + d)$$

$$T_3 = f_B(t_3), T_4 = f_s(t_3 + d)$$

# The Challenges of High Latency (Cont'd)

- No clock skew

$$f_S(t) = t + b_S$$

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

$$\widehat{f}_S(t) = f_S(t) + \beta_S(t)$$

# The Challenges of High Latency (Cont'd)

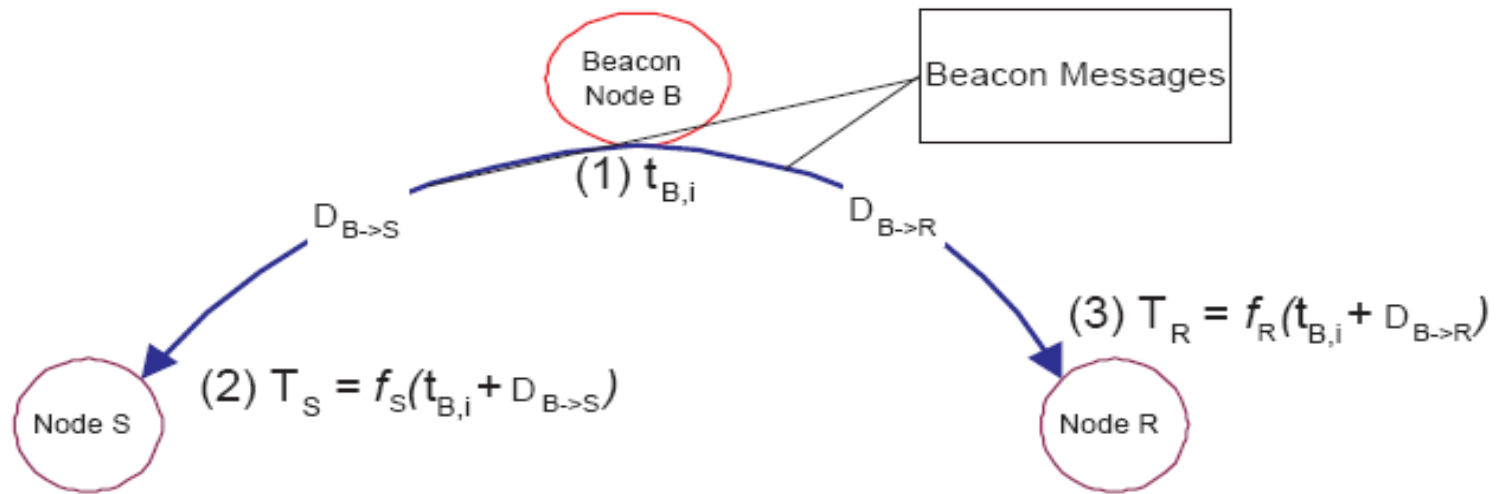
- Effect of clock skew

$$f_S(t) = at + b_S$$
$$\beta(t_3 + d) = \frac{(1 - a_S)(t_1 + t_3 + d) - 2b_S}{2}$$

- Error

$$Error = \frac{(1 - a_S)((t_3 - t_1) + d)}{2}$$

# Skew Synchronization



$f_R(t), f_S(t)$  represent the individual, unsynchronized clocks of nodes R & S.

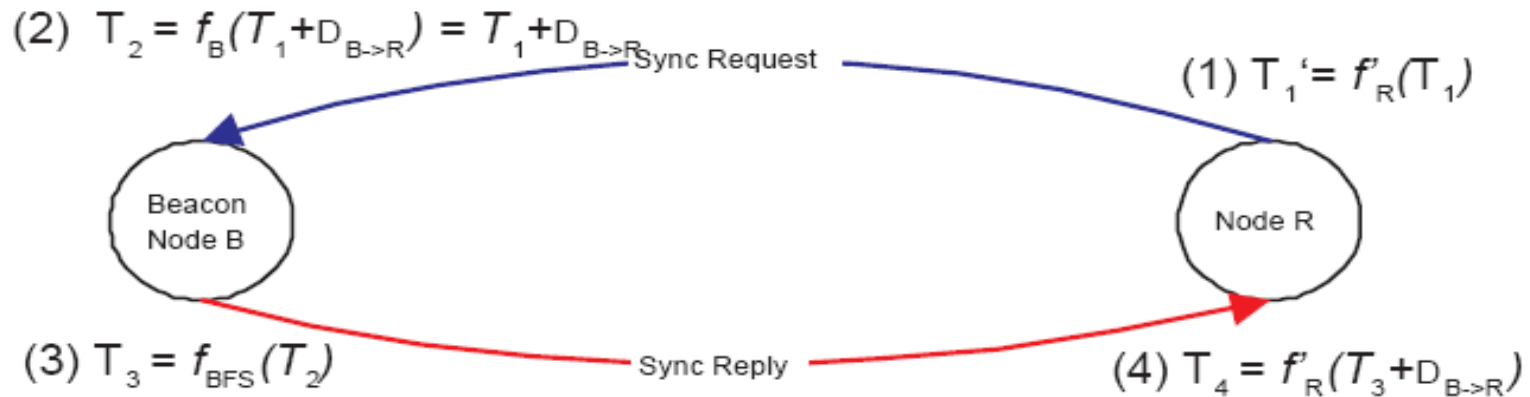
**Phase 1**

# Skew Synchronization (Cont'd)

- Linear regression the data points

$$(t_{B,i} - f_R(t_{B,i} + D_{B \rightarrow R}), f_R(t_{B,i} + D_{B \rightarrow R}))$$

# Offset Synchronization



$f'_R(t)$  represents skew-corrected clock, calculated in Phase 1. Note that events are ordered, in time, based on their index.

## Phase 2

# Offset Synchronization (Cont'd)

$$\text{offset}_R = \left[ \left( f_B(f'_R(T1) + D_{R \rightarrow B}) - f'_R(T1) \right) - \left( f'_R(T3 + B \rightarrow R) - T3 \right) \right] / 2$$

