Ch 10 - Goals

- Introducing the synchronous abstraction
- Simulating the synchrnous abstraction
- Approximate agreement
- Pulse synchronization with better skew

Approximate Agreement

- Each node $v \in V_g$ is given and input $r_v \in \mathcal{R}$. Given $\mathcal{E} > 0$. Generate output value $o_v \in \mathcal{R}$, such that
  - agreement: $\max_{v,w \in Vg}\{|o_v - o_w|\} \leq \mathcal{E}$
  - validity: for each $v \in V_g$ is $\min_{w \in Vg}\{r_w\} \leq o_v \leq \max_{w \in Vg}\{r_w\}$
  - termination: each node $v \in V_g$ outputs its value, o_v, and terminate within finite number of rounds.

- Thus, nodes need to exchange their input values and try (iteratively) to compute output values that satisfy the above requirement.
- Faulty nodes may send arbitrary values to try to prevent the convergence of output values.

Approximate Agreement Algorithm

The algorithm proceeds in rounds.

The input to each round is the output of the previous round.

The initial input is the input of the first round.

The Basic Iteration:

1. send r_v to all.
2. receive $r_{w,v}$, the value sent by w in this round.
   // replace any "missing" value by r_v
3. $S_v := \{r_{w,v}\}$; //ordered set
4. $o_v := (S_v^{(f+1)} + S_v^{(n-f)})/2$; // the (f+1)st and (n-f)-th values in S
5. Return o_v

The initial range is reduced by half at the end of each iteration

Insights

The approximate agreement **approach** enables correct nodes to reduce the range of values they hold at the beginning of an iteration, despite the Byzantine behavior.

Therefore, one can use the approach so if the values that correct nodes hold drifts apart (the range expands), we can occasionally run the basic iteration to converge back to a desired range of values.

Alternatively, we can run the iteration so the upper bound on the range of values will not exceed some maximal value, by regularly reducing the spread of values.

If the initial range is large, we can run full approximate agreement to reduce the initial range.

Improving pulse skew

We will use the idea of the approximate agreement to present an algorithm that achieves better skew than what is obtained in the previous chapter.

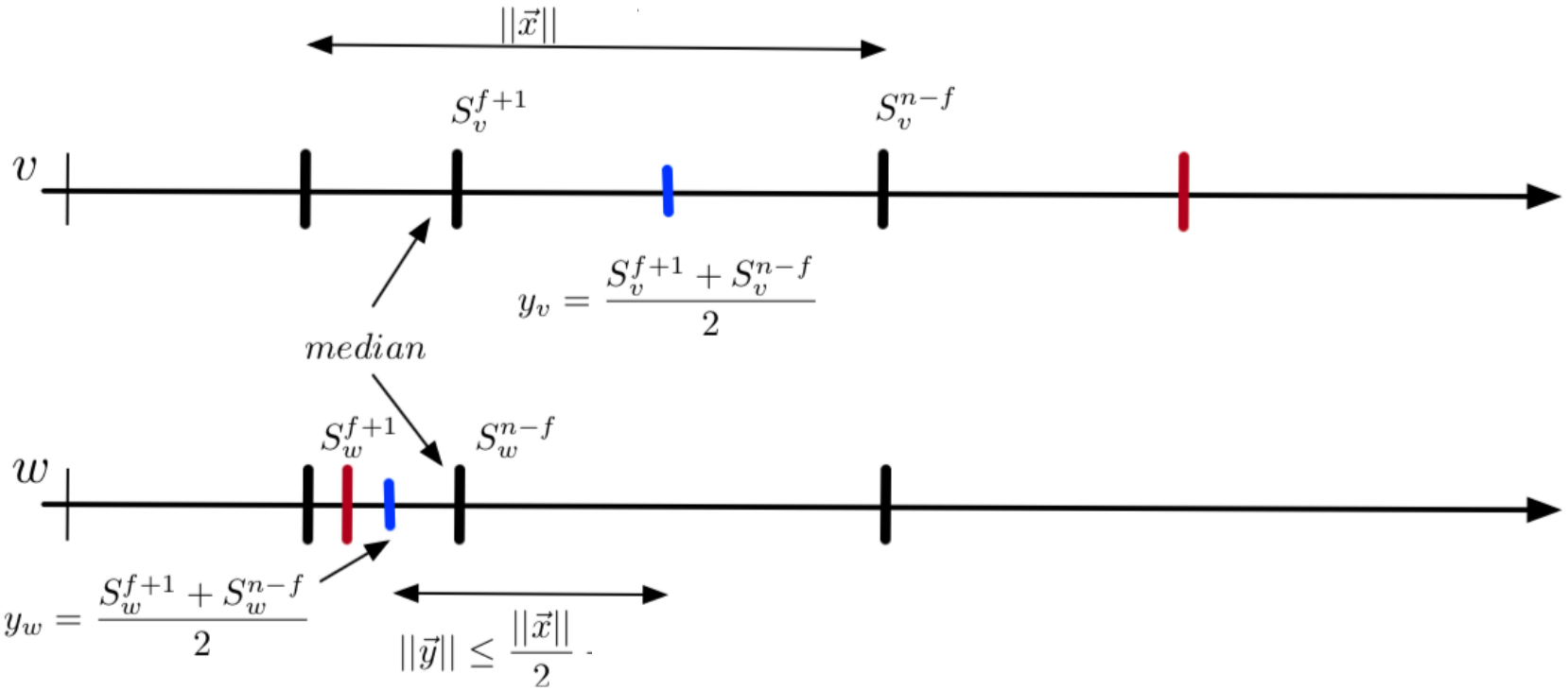
We will not exchange specific values, as we do in the approximate agreement. Instead messages will be sent at a fixed amount of time past the beginning of each round, and from the reception of the messages nodes will try to determine the clock value at the sender's node when it began its round and invoked its pulse.

There is uncertainty about the current clock value that stems from the initial skew, transmission time and clock drift.

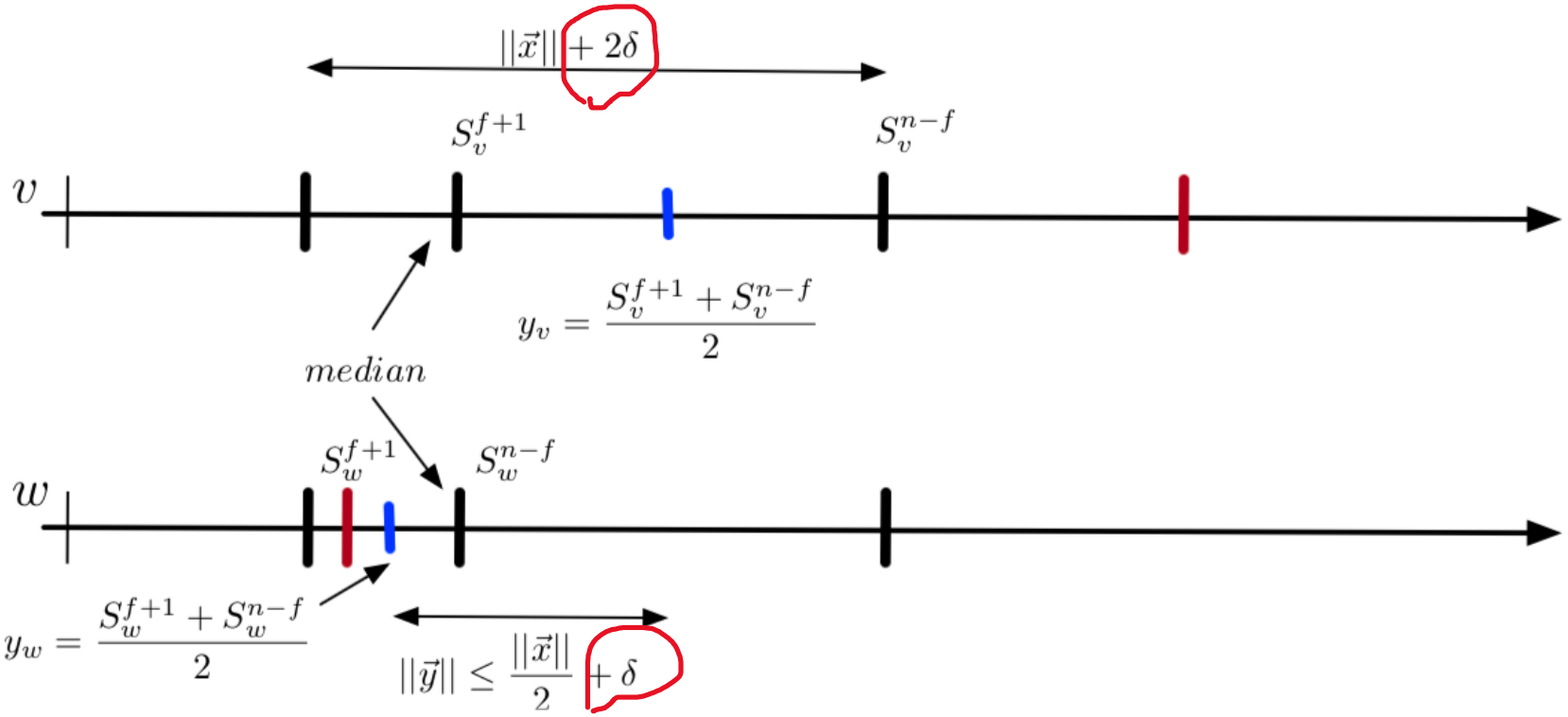
i.e, instead of r_w we get, r_w' where, $r_w \leq r_w' \leq r_w + \delta$

where δ is the upper bound on the error of the estimates of other correct nodes' offsets.

4 nodes – 1 faulty



4 nodes – 1 faulty and perturbation in values



Corollary 10.5

Let δ be the upper bound on the neighbors clock estimates at a correct node. The approximate agreement algorithm obtain the following:

For each $v \in V_g$ is $r_g^{(1)} \leq o_v \leq r_g^{(n-f)} + \delta$

In each iteration, $[\![o]\!] \leq [\![r_g]\!]/2 + \delta$

Basic Parameters

The algorithm we present assumes known n, f, d and ϑ.

$p_{v,r}$ is the real time at which $v \in V_g$ invokes its r-th pulse.

Nodes regularly invoke pulses, collect estimates of other nodes pulse invocation times and use the approximate agreement approach to estimate the desired correction to the time at which to invoke the next pulse.

S – is an upper bound of the initial hardware clock values, and we determine it value so it will also be the upper bound on the skew of pulses.

T – the nominal round duration between pulses and between approximate agreement iterations

δ – is the upper bound on the error of the estimates of other correct nodes' offsets

Insights (again)

The approximate agreement **approach** enables correct nodes to reduce the range of values they hold at the beginning of an iteration, despite the Byzantine behavior.

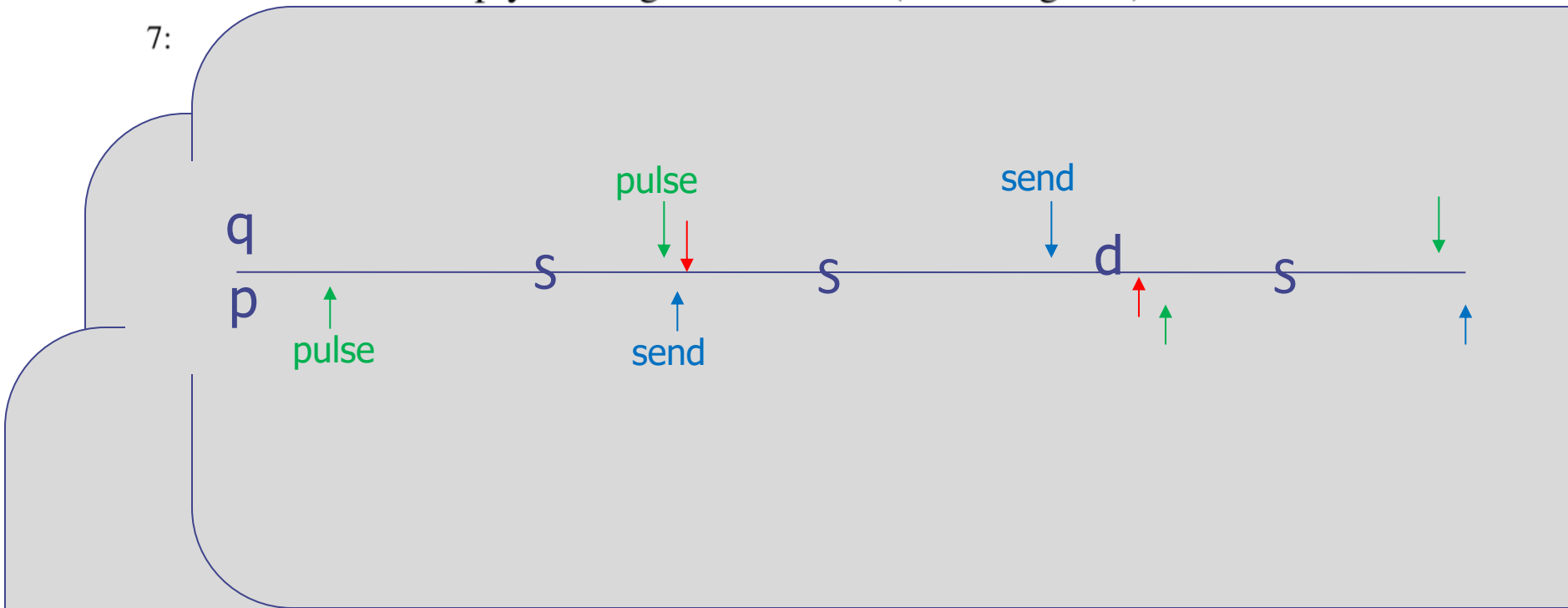
Therefore, one can use the approach so if the values that correct nodes hold drifts apart (the range expands), we can occasionally run the basic iteration to converge back to a desired range of values.

Alternatively, we can run the iteration so the upper bound on the range of values will not exceed some maximal value, by regularly reducing the spread of values.

If the initial range is large, we can run full approximate agreement to reduce the initial range.

Algorithm 10 Lynch-Welch pulse synchronization algorithm, the code for a node $v \in V_g$. $\mathcal{S}$ denotes a (to-be-determined) upper bound on $\|\vec{p}_r\|$ for each $r \in \mathbb{N}_{>0}$ and T is the nominal round duration and it needs to be specified how Line 9 is implemented.

1: wait until getH() $= \mathcal{S}$ // $H_w(0) \in [0, \mathcal{S})$ for all $w \in V$
2: **for all** round $r \in \mathbb{N}$ **do**
3: generate r-th pulse
4: $h \leftarrow$ getH()
5: wait until getH() $= h + \vartheta\mathcal{S}$ // all nodes are in round r
6: broadcast empty message to all nodes (including self)
7:



Algorithm 10 Lynch-Welch pulse synchronization algorithm, the code for a node $v \in V_g$. $\mathcal{S}$ denotes a (to-be-determined) upper bound on $\|\vec{p}_r\|$ for each $r \in \mathbb{N}_{>0}$ and T is the nominal round duration and it needs to be specified how Line 9 is implemented.

1: wait until getH() $= \mathcal{S}$ // $H_w(0) \in [0, \mathcal{S})$ for all $w \in V$
2: **for all** round $r \in \mathbb{N}$ **do**
3: generate r-th pulse
4: $h \leftarrow$ getH()
5: wait until getH() $= h + \vartheta\mathcal{S}$ // all nodes are in round r
6: broadcast empty message to all nodes (including self)
7: wait until getH() $= h + (\vartheta^2 + \vartheta)\mathcal{S} + \vartheta d$ // denote this time by $\tau_{v,r}$
// correct nodes' messages should have arrived
8: **for** each node $w \in V$ **do**
9: compute $\Delta(w) \in [p_{w,r} - p_{v,r}, p_{w,r} - p_{v,r} + \delta]$
// denote $p_r := \max_{w \in V_g}\{p_{w,r}\}$
10: **end for**
11: $S \leftarrow \{\Delta(w) \mid w \in V\}$ (as multiset, i.e., values may repeat)
12: $\Delta \leftarrow \left(S_v^{(f+1)} + S_v^{(n-f)}\right)/2$
13: wait until getH() $= h + \Delta + T$
14: **end for**

Initial requirements on round execution

Since we simulate an approximate agreement in a synchronous environment we need to make sure that the following holds:

1) Messages sent by correct nodes in a given round should be received by all correct nodes after they start the
✓ current round and before they compute the clock estimates, i.e., during $[p_{v,r}, \tau_{v,r}]$

2) T is large enough to accommodate the adjustments for the next iteration, i.e., $H_v(\tau_{v,r}) \leq H_v(p_{v,r}) + \Delta + T$

We prove by induction that both hold and that $[\![p_r]\!] \leq S$

As a base case, we use the bound on the skew of the first round

Basic Inductive Step

Lemma 10.8. *Suppose that* $T \geq (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d$ *and*

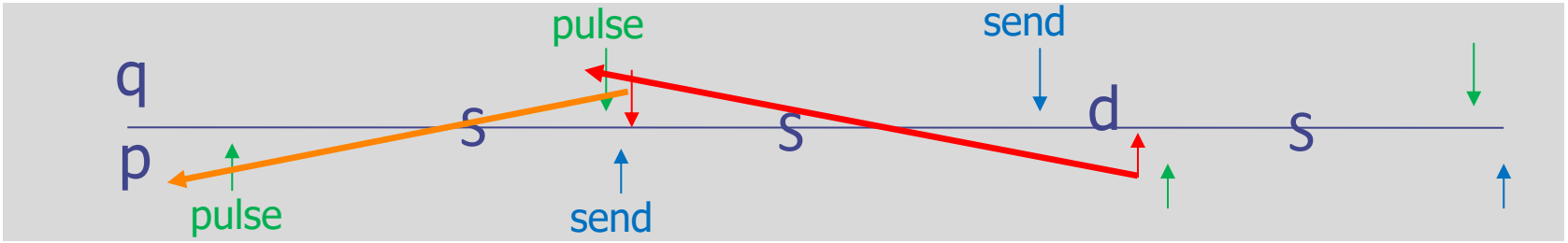
$$\mathcal{S} \geq \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta}.$$

T > 3S + d

S > 2(δ + drift(T))

Moreover, assume that for $r \in \mathbb{N}$ *it holds that all prior rounds have been executed correctly, and that* $\|\vec{p}_r\| \leq \mathcal{S}$. *Then*

(i) round r is executed correctly,



We will choose $\Delta(w) := H_v(t) - S - (d-u) - H_v(p_{v,r})$

We can prove that $-S \leq \Delta_v \leq S + \delta$

δ will be chosen to satisfy our assumptions on estimates

Basic Inductive Step

Lemma 10.8. *Suppose that* $T \geq (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d$ *and*

$$\mathcal{S} \geq \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta}.$$

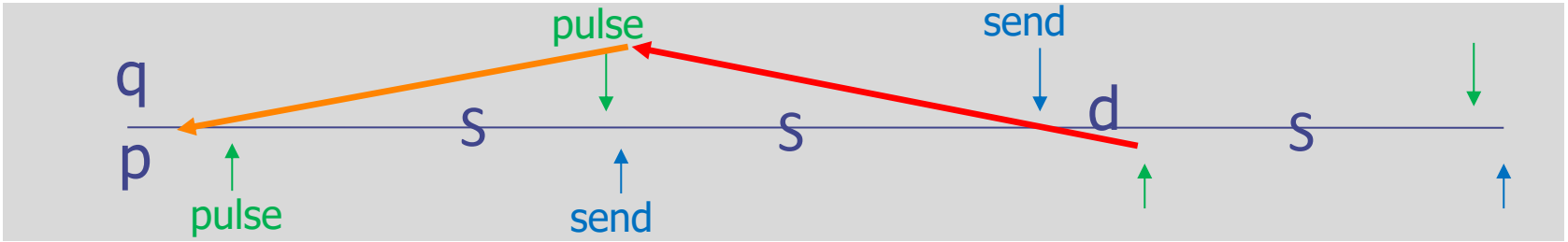
T > 3S + d

S > 2δ + drift(T)

Moreover, assume that for $r \in \mathbb{N}$ *it holds that all prior rounds have been executed correctly, and that* $\|\vec{p}_r\| \leq \mathcal{S}$. *Then*

(i) round r is executed correctly,

(ii) $(T - \mathcal{S})/\vartheta \leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} \leq T + \mathcal{S} + \delta$, *and*



Observe that from $-S \leq \Delta_v \leq S + \delta$

$$H_v(p_{v,r+1}) - H_v(p_{v,r}) = T + \Delta_v \geq T - S$$

Therefore, $p_{v,r+1} - p_{v,r} \geq (T - S) / \vartheta$

Basic Inductive Step

Lemma 10.8. *Suppose that* $T \geq (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d$ *and*

$$\mathcal{S} \geq \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta}.$$

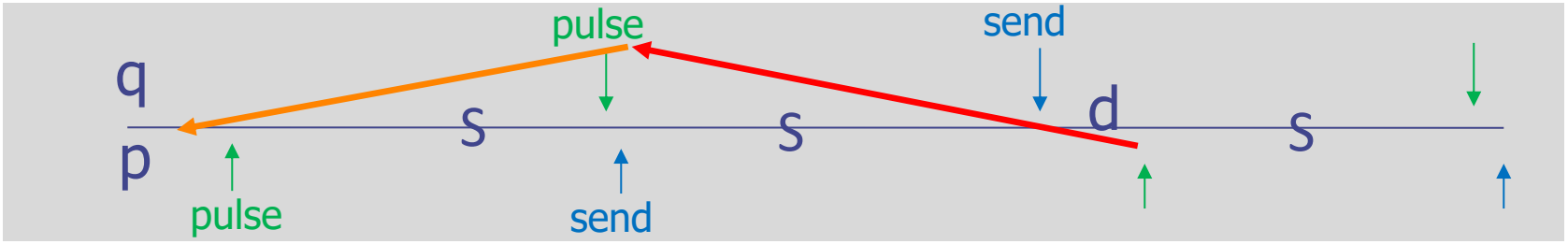
T > 3S + d

S > 2δ + drift(T)

Moreover, assume that for $r \in \mathbb{N}$ *it holds that all prior rounds have been executed correctly, and that* $\|\vec{p}_r\| \leq \mathcal{S}$. *Then*

(i) round r is executed correctly,

(ii) $(T - \mathcal{S})/\vartheta \leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} \leq T + \mathcal{S} + \delta$, *and*



Observe that from $-S \leq \Delta_v \leq S + \delta$

$$H_v(p_{v,r+1}) - H_v(p_{v,r}) = T + \Delta_v \leq T + S + \delta$$

from which we conclude the right hand side.

Basic Inductive Step

Lemma 10.8. *Suppose that* $T \geq (\vartheta^2 + \vartheta + 1)\mathcal{S} + \vartheta d$ *and*

$$\mathcal{S} \geq \frac{2(2\vartheta - 1)\delta + 2(\vartheta - 1)T}{2 - \vartheta}.$$

T > 3S + d

S > 2(δ + drift(T))

Moreover, assume that for $r \in \mathbb{N}$ *it holds that all prior rounds have been executed correctly, and that* $\|\vec{p}_r\| \leq \mathcal{S}$. *Then*

(i) round r is executed correctly,

(ii) $(T - \mathcal{S})/\vartheta \leq \min_{v \in V_g}\{p_{v,r+1}\} - \min_{v \in V_g}\{p_{v,r}\} \leq T + \mathcal{S} + \delta$, *and*

(iii) $\|\vec{p}_{r+1}\| \leq \mathcal{S}$.

$p_{v,r+1} - p_{w,r+1} \leq p_{v,r} + \Delta_v + T - (p_{w,r} + (\Delta_w + T)/\vartheta) =$

$p_{v,r} + \Delta_v - (p_{w,r} + \Delta_w) + (1-1/\vartheta)(\Delta_w + T)$

$\leq$ $S/2 + \delta + (1-1/\vartheta)(\Delta_w + T)$

$\leq$ $S/2 + \delta + (1-1/\vartheta)(T + S + \delta)$

Choosing S as stated in the lemma completes the proof

The Final Claim

Theorem 10.10. *Assume that* $6 - 2\vartheta - \vartheta^2 - 2\vartheta^3 > 0$ *and that estimates are computed according to Lemma 10.9. For any choice of*

$$T \geq \frac{(4\vartheta^3 + 2\vartheta^2 + 2\vartheta - 2)u + (4\vartheta^4 - 3\vartheta^3 - 2\vartheta^2 + 2)d}{6 - 2\vartheta - \vartheta^2 - 2\vartheta^3} \in O(d),$$

set

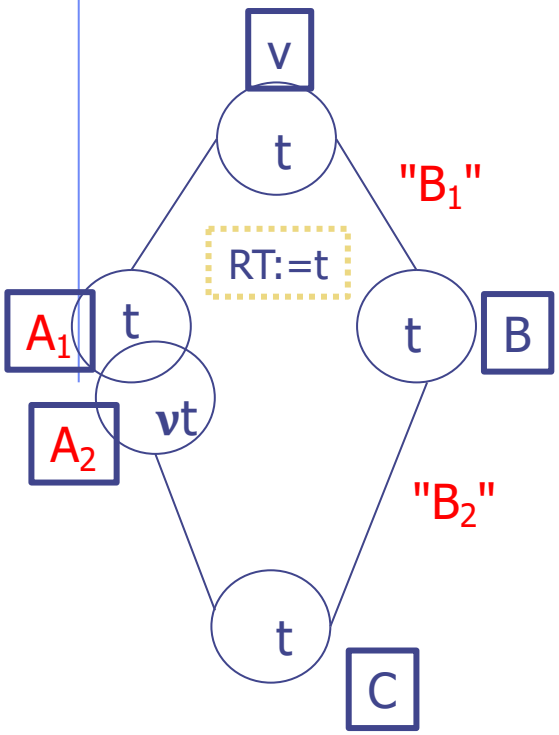
$$\mathcal{S} := \frac{2(2\vartheta - 1)(u + (\vartheta - 1)d) + 2(\vartheta - 1)T}{4 - 2\vartheta - \vartheta^2} \in O\left(u + \left(1 - \frac{1}{\vartheta}\right)T\right).$$

If $\max_{v \in V}\{H_v(0)\} \leq \mathcal{S}$, *then Algorithm 10 solves pulse synchronization with skew at most* $\mathcal{S}$, $P_{\min} \geq (T - (\vartheta + 1)\mathcal{S})/\vartheta$, *and* $P_{\max} \leq T + 3\mathcal{S}$.

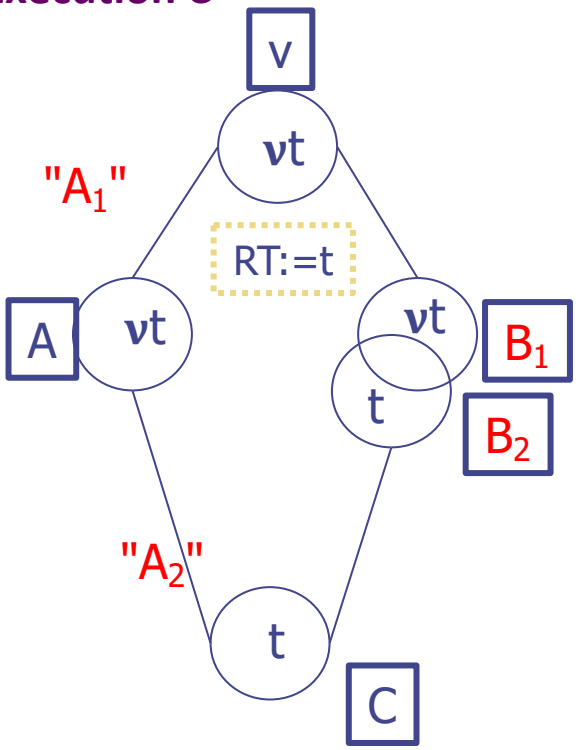
The proof is straightforward, using the inductive claim and the choices of the parameters.

No Pulse Synch Algorithm for small outdegree

Execution $\mathcal{E}$



Execution $\mathcal{E}'$



Correct nodes can't tell the difference.
v and C fire their pulses at the same clock times in both $\mathcal{E}$ and $\mathcal{E}'$.
Skew increases over time.