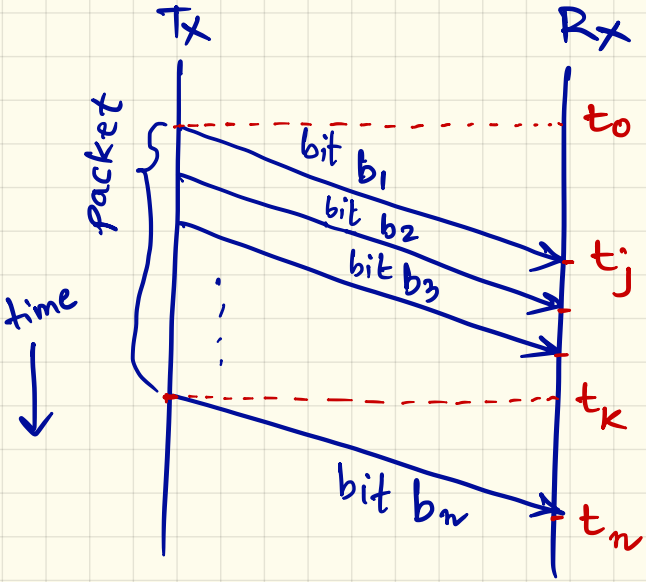
# MAC

## ⊖ Propagation Delay and Transmit time :

Question: How long does it take for a transmitter to fully transmit a packet to the receiver ?




Propagation delay (PD) $= (t_j - t_0) = \dfrac{d_{TR}}{c}$

speed of light

Transmit time (TT) $= (t_k - t_0) = (t_n - t_j)$

$= \dfrac{L}{R}$

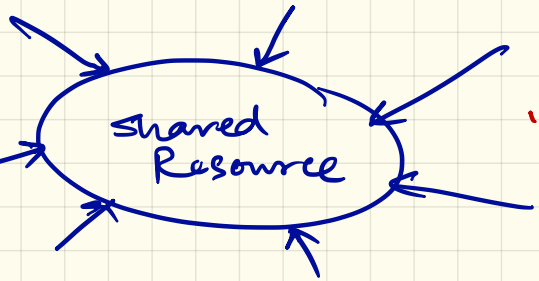
$L \longrightarrow$ size of pkt in bits

$R \longrightarrow$ Rate of transmission

Total time $= PD + TT$

# ③ Medium Access Control (MAC)

How to share a resource in a distributed manner?



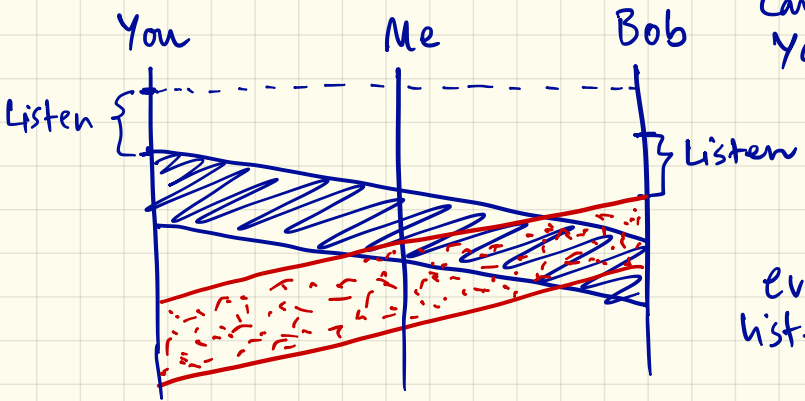

Resource is the "channel"

**Idea 1**: I want to transmit → Just transmit

**Problem 1**: But all transmitters will collide.

**Idea 2**: Listen before you talk to ensure no one else transmitting

**Problems 2**
- simultaneous transmission cannot be avoided.
- Propagation delay problems, i.e., Bob cannot hear you until propagation delay. Can talk before that even after listening.

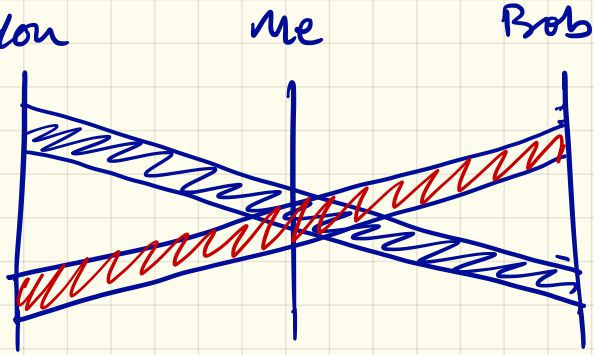


**Idea 3**

Listen before talking and also while talking. ⇒ If some foreign signal is heard, conclude there is a collision ⇒ Prepare to retransmit

**Problem 3**

- Collision still possible.



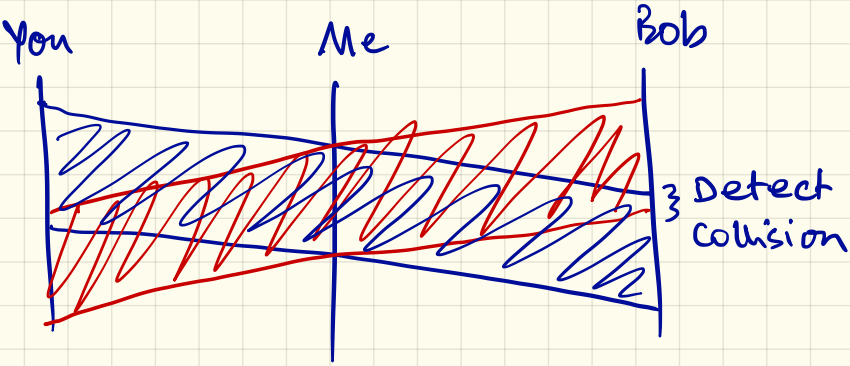

**Idea 4**

Make pkt size longer, i.e.,

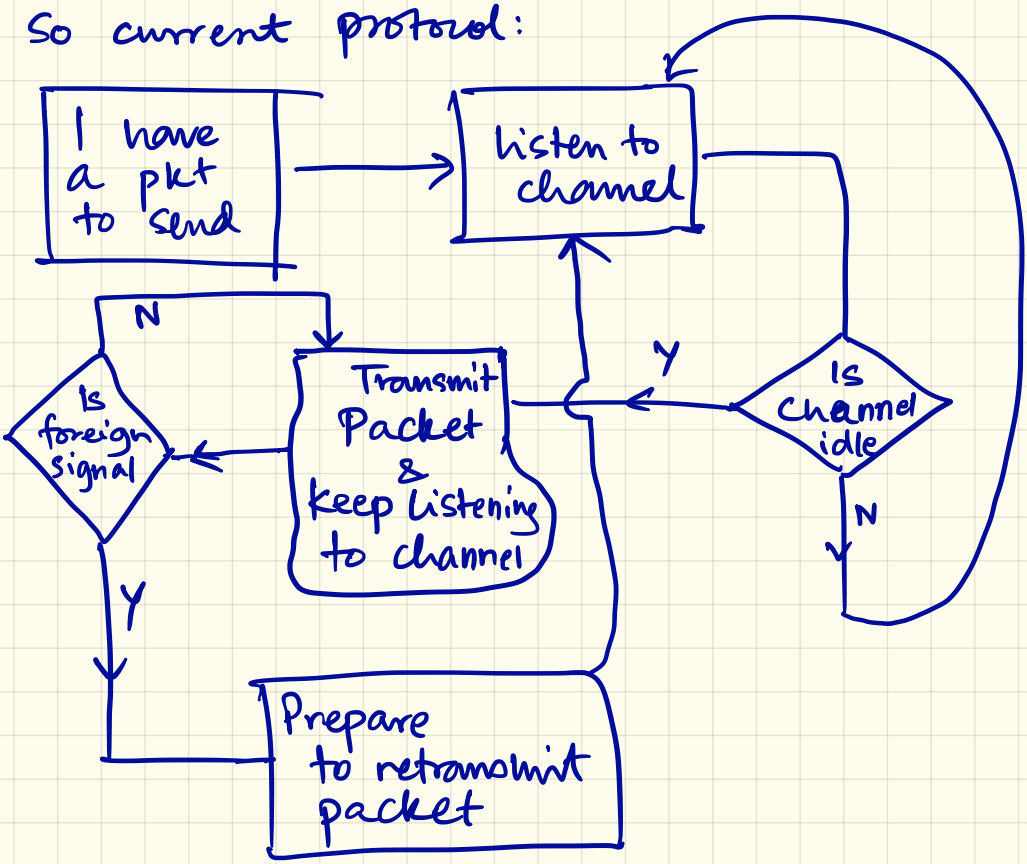
Transmit time > Max network prop. delay.

OR

Require ACKs from the Receiver.



So current protocol:

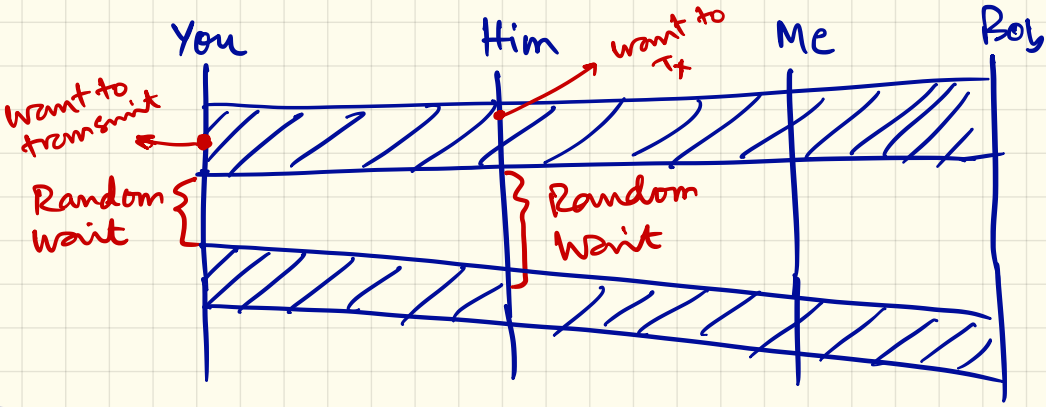



**Problem 4**

- Many nodes want to transmit. But all of them experience busy channel. So they all wait for channel to become idle. Then they all transmit ⇒ collision for sure.

**Idea 5**

Wait for random times after channel becomes idle.



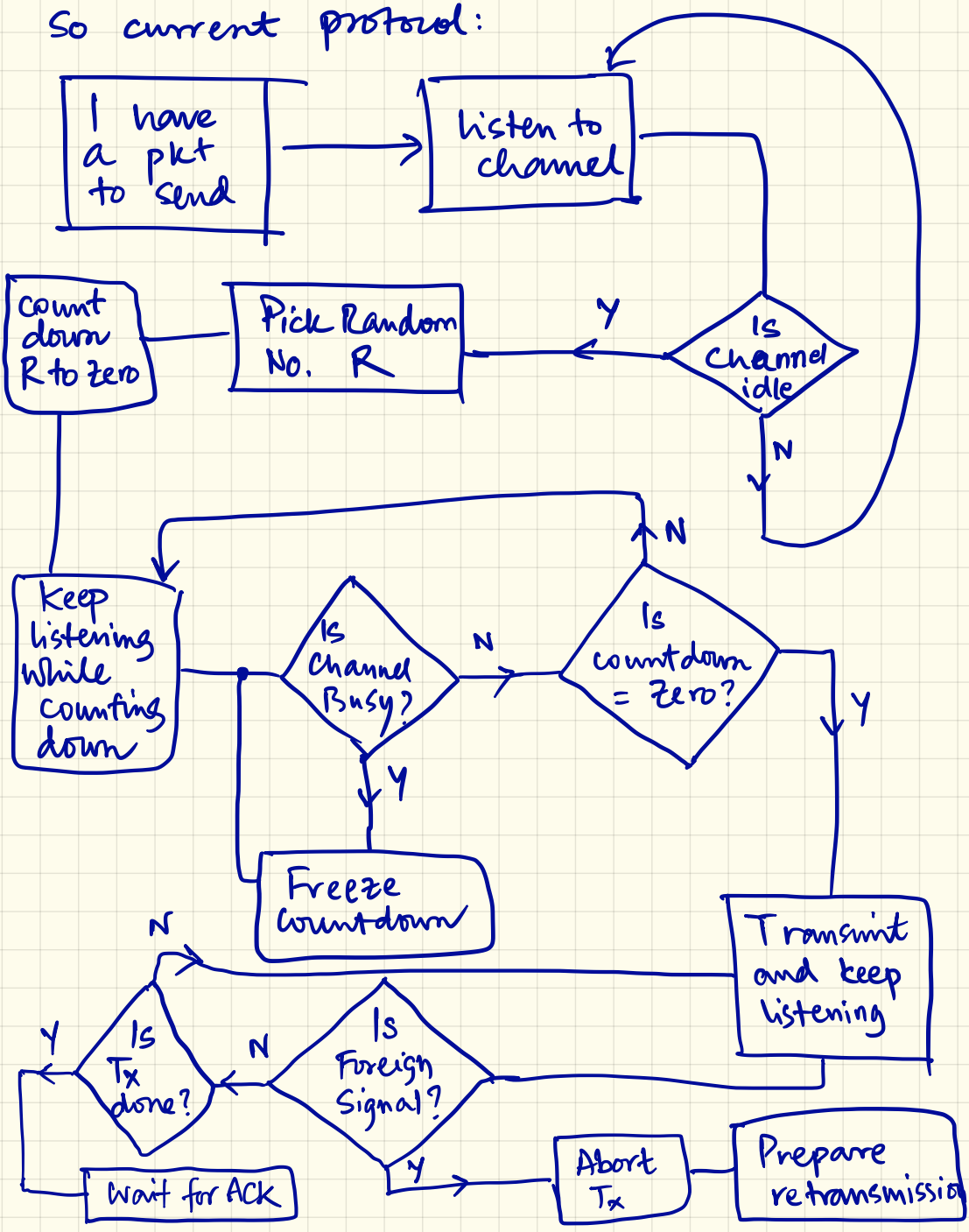

**Problem 5**

- Node may receive unfair share of channel just due to bad luck. Starvation possible (although low probability).

**Idea 6**

Don't reset random no. if unable to transmit. Insted freeze the random count down once channel becomes busy. Resume countdown once channel idle again.

So current protocol:



**Problem 6**

- How do you cope with collision? Do you pick new random # and try again? Can there be collision again?

**Idea 7**

Increase random number range after collision. Pick new rand #.

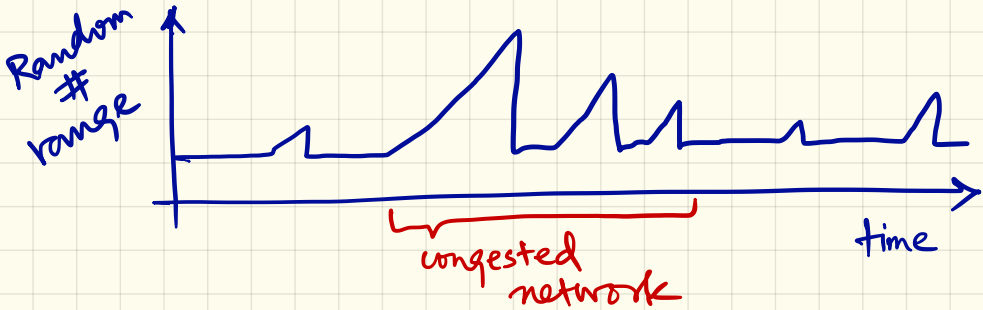
This is a form of admission control, i.e., reduce the rate of pkt injection into the network.

**Problem 7**

- But congestion may have gone and you still pick large random #s.

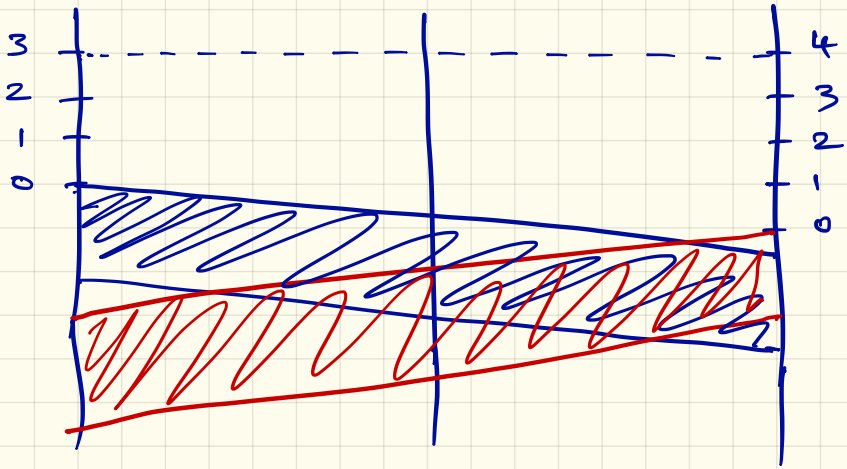
**Idea 8**

Once pkt is successful (i.e., no congestion) then reset your random # range back to minimum value.



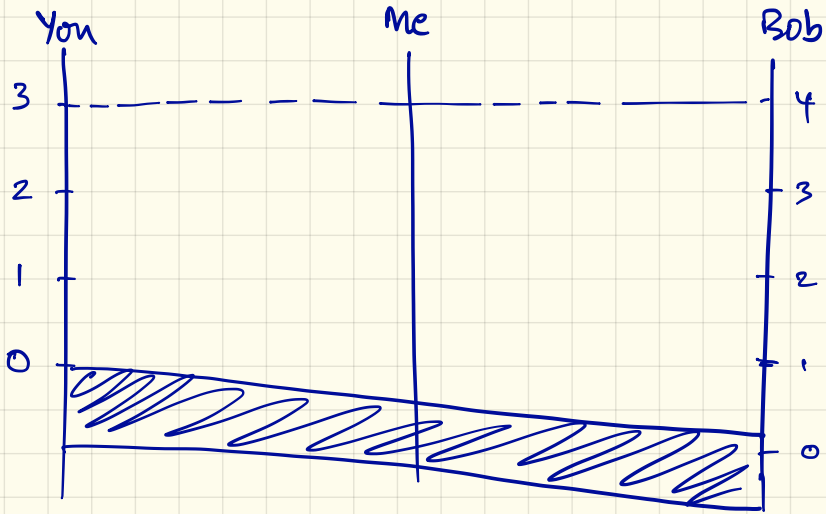
**Problem 8**

- Collision still possible even if 2 nodes have picked different random #s.



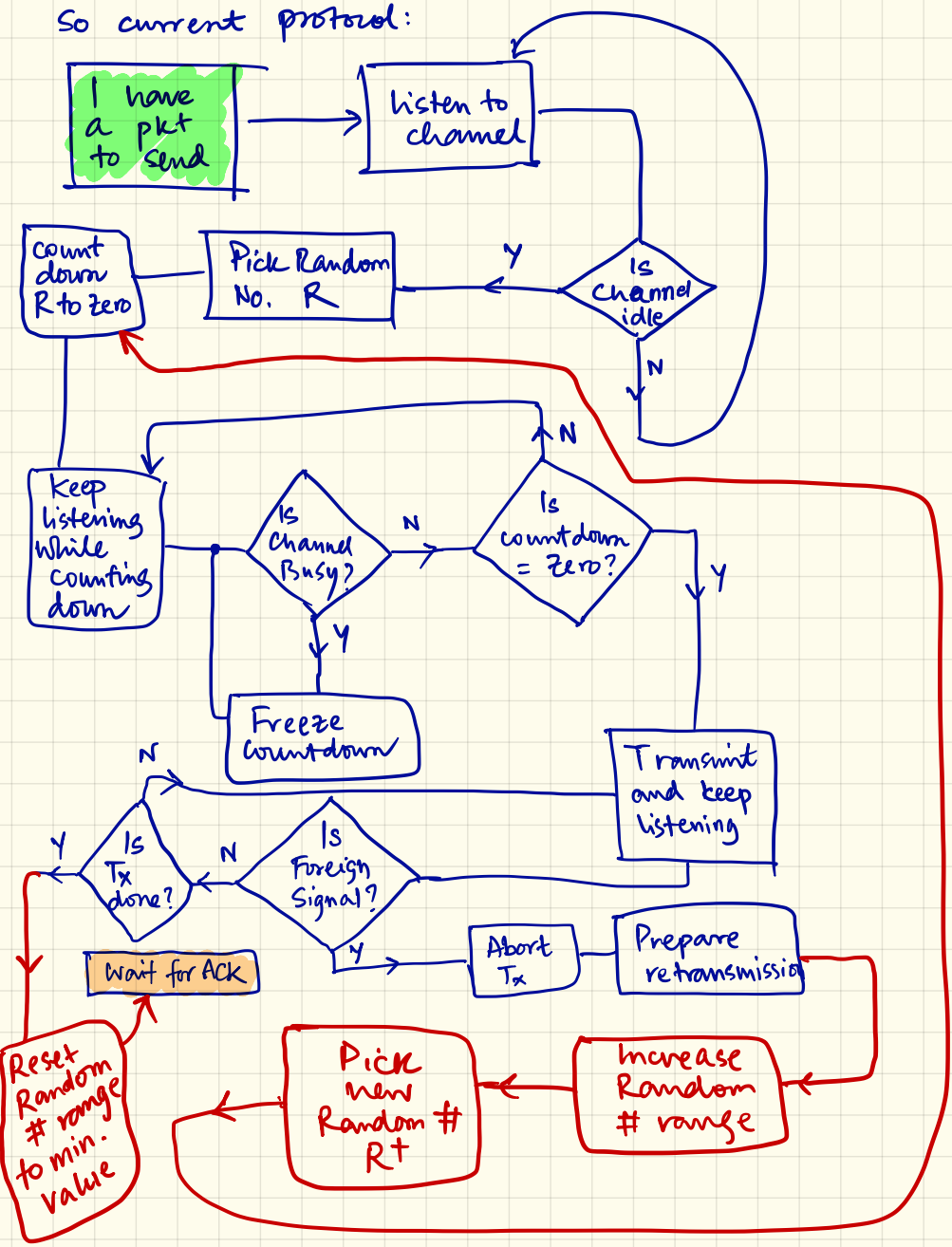

**Idea 9**

Make the countdown as multiples of slots, where each slot time is greater than prop. delay.




Bob doesn't transmit. NO collision

So current protocol:



(→) Some key points.

(1) Collision happens always at the Receiver. Transmitter may detect collision by observing a foreign signal, but that doesn't mean collision is at $T_x$.

(2) Channel is wasted because of random count down ⇒ called BACKOFF. This is the price to be paid for distributed coordination.

(3) The above protocol assumes that a $T_x$ can transmit and listen at the same time. Possible in wired networks like Ethernet. Harder in wireless networks.

(4) $T_x$ detects foreign signal and can tell for sure that collision is happening at $R_x$. This assumes channel is identical at $T_x$ and $R_x$. True for wired networks, not for wireless.